ISSN 1870-4069

# How to Learn Picture Languages

David Kuboň, František Mráz

Charles University, Prague, Czech Republic
{dkubon,mraz}@ksvi.mff.cuni.cz

**Abstract.** Analysis of sentences in a natural language is often based on similar methods as analysis of formal languages. Analogically, analysis of pictures could be based on analysis of formal picture languages. However, the field of formal picture languages is not developed enough for this purpose. This paper presents several models of automata accepting two-dimensional languages and outlines their learning capabilities. Further, it examines the possibility of transforming a two-dimensional language into a one-dimensional language and applying machine learning techniques in a single dimension. In this paper, we propose a new representation for formal picture languages consisting of two components – a picture-to-string function and a string language. The function rewrites any two-dimensional picture into a string. A picture language is then the set of all pictures that the function maps into the given string language. Using this representation, picture languages can be learned by applying methods of grammatical inference for string languages.

**Keywords:** learning, grammatical inference, automata, formal languages, picture languages.

## 1 Introduction

In comparison to one-dimensional (string) languages, our knowledge about two-dimensional (picture) languages is limited [14], even though their theoretical and practical significance is comparable – they can be both used as formal models of practical problems. In the case of picture languages it can be for instance automatic detection of different shapes (e.g. road signs) or more generally any problem on two-dimensional data which has some pattern regularity [30].

In this work we focus on formal two-dimensional languages, i.e. sets of two-dimensional pictures that have formally exact description. They are referred to as *picture languages*, but they are not sets of pictures in the common sense, as such sets, like the set of photos containing cars, cannot be defined mathematically. While deep neural networks are known as the best tools for recognizing objects in images [19], their application for picture languages is limited. On the other hand, powerful models of automata working on two-dimensional inputs are usually non-deterministic and inefficient for applications.

Here we propose a method for learning picture languages from positive and negative samples. Learning a model (a grammar) for a target language based on some information about the words of the language is called *grammatical* *inference* [15]. In case of one-dimensional (string) languages, there are several known algorithms of grammatical inference for a number of classes of languages. Much less is known about grammatical inference for two-dimensional languages.

Two types of representations of pictures can be found in the literature. The first one is generative – such representation describes how a picture can be generated from a string. Freeman in [12] introduced an 8-letter alphabet interpreted as movements north, south, east, west, northeast, southeast, northwest, and southwest. By interpreting a word over the alphabet called a "chain code" we obtain a drawing. Hence, a picture is a set of unit length lines on a plane. This was later simplified to a 4-letter alphabet with the first four movements from the Freeman's alphabet [23]. Later, Costagliola [11] extended the alphabet in order to generate colored pictures or pictures with labels.

The second representation of a picture is a rectangular array of symbols that can be interpreted as colors of pixels in the image.

Using the first representation of pictures, a picture language can be represented as a set of strings describing all pictures in the picture language. Using the second representation of pictures, a picture language is the set of pictures accepted by a device (automaton) working on two-dimensional inputs. For example, the class of recognizable picture languages is accepted by non-deterministic online tessellation automata [14], even more powerful are sgraffito automata [33] and two-dimensional limited context restarting automata [20]. These automata models are quite powerful but they share high complexity – the problem whether given input picture is accepted by such automaton is NP-complete.

Here we propose a new representation for picture languages, which uses rectangular arrays of symbols for representing pictures and string languages for representing sets of pictures. The new representation consists of a function $R$ that rewrites any two-dimensional picture $p$ into a string $R(p)$ and a one-dimensional (string) language $L$. The picture language is then the set of all pictures $p$ for which $R(p)$ is in $L$. Further we propose one such function $R$ and combine it with a classical algorithm for inferring regular languages from positive and negative samples. In this way we obtain a method for learning picture languages.

The paper is structured as follows. After introducing basic definitions for pictures and picture languages in Section 2, the next section introduces the used learning paradigm. Then, in Section 4 we discuss the generative representation of picture languages and some known results on learning picture languages represented in this way. Next, in Section 5 we informally present several models of two-dimensional automata which work on pictures as rectangular arrays of symbols. Also some results on learning such automata are included. The crucial Section 6 introduces the new representation of picture languages and lists results of experiments with inferring representations for several simple picture languages using grammatical inference for regular languages. Concluding section contains outline for further research.

## 2 Pictures and Picture Languages

In the past, several authors recognized that describing multidimensional objects by strings enables them to apply the means developed in the field of formal languages for studying sets of objects [23]. The first approach to describe two-dimensional pictures by one-dimensional strings were chain codes. Here we use its simplified version from [23]. According to it, a picture is a set of unit length lines in the Cartesian plane. A word in the alphabet $\Pi = \{l, r, u, d\}$ is a *picture description.* A set of picture descriptions is a *picture description language.* For a picture description $q \in \Pi^*$, its interpretation denoted as $pic(q)$ is the drawing obtained in the following way: The interpretation starts by placing a pen at any point with integer coordinates in the Cartesian plane. Next each letter $x$ of $q$, from left to right, is interpreted as moving the pen by unit distance in the direction left, for $x = l$, right, for $x = r$, up, for $x = u$, and down, for $x = d$. E.g., the letter 'L' can be drawn as $uuddr$. Obviously, the resulting picture is connected. A picture consisting of several non-connected parts would be possible to represent by extending the alphabet $\Pi$ by symbols $\uparrow$ and $\downarrow$, for lifting and lowering the pen above and down to the drawing plane, respectively.

Of course, such interpretation can draw a picture at any position in the plane. We usually do not distinguish between pictures which differ by their position in the Cartesian plane – the detailed definitions can be found in [23].

For a nonempty fixed picture $q$ consisting of a connected set of unit length horizontal and vertical lines there exists infinite number of its picture descriptions. Consider for example the set of picture descriptions given by the regular expression $(rl)^+$ that all describe a horizontal line of unit length. The set of all picture descriptions of $q$ is denoted as $des(q)$. It is known that $des(q)$ is a regular language [23].

The second approach defines a *picture* $P$ as a two-dimensional rectangular array of elements from a finite alphabet $\Sigma$ (see [13]). We say that $P$ has dimensions $(m, n)$, if it has $m$ rows and $n$ columns. Then $P_{i,j}$ from $\Sigma$ denotes the symbol at position $j$ in row $i$. The set of all rectangular pictures over $\Sigma$ of dimensions $(m, n)$ will be denoted as $\Sigma^{m,n}$ and the set of all rectangular pictures over $\Sigma$ of any dimension will be denoted as $\Sigma^{*,*}$. A *picture language* is then any subset of $\Sigma^{*,*}$.

Any automaton working on an picture $P$ of dimensions $(m, n)$ needs to know where is the border of the picture, therefore the picture is usually surrounded by sentinels #, where $\# \notin \Sigma$. Delimited picture $P$ is called *boundary picture* $\widehat{P}$ over $\Sigma \cup \{\#\}$ of dimensions $(m + 2) \times (n + 2)$ – see Fig. 1.

## 3 The Learning Paradigm of Exact Identification

The learning paradigm of exact identification [3] has already been applied in a number of domains including regular [2] and context-free languages [34].

A *concept* in a universe of objects is any subset of the universe. A *class of concepts* is a set of concepts. Concepts can be usually represented by words

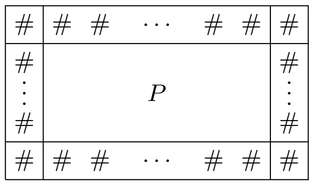

**Fig. 1.** The boundary picture $\widehat{P}$.

over some fixed finite alphabet. Then, the learning lies in exactly identifying an unknown (target) concept chosen from a specified class of concepts $\mathcal{C}$, i.e. computing a representation of the unknown concept. To learn, the learner may use some given information about the target concept (for example, a set of positive and negative samples, i.e. words from the concept and words not belonging to the concept) or it can even pose certain types of queries to a teacher returning the correct answers. Usually, the following two types of queries are considered (see [2]):

– a membership query, where the learner proposes an object $x$: the reply is *yes* if $x$ belongs to the target concept and *no* otherwise, and
– an equivalence query, where the learner proposes a representation $p$ (in the specified system) of a concept in $\mathcal{C}$: the reply is either *yes* if $p$ represents the unknown concept or *no* if $p$ is wrong and in this case the teacher also returns an arbitrary object $x$ that $p$ and the target concept classify differently.

Any class of concepts is trivial to learn with equivalence queries given a recursively enumerable set of representations: we just ask until the *yes* answer is received; and thus the main interest lies in the efficiency of learning algorithms.

Several known methods (protocols) can be used for learning (string) languages (see [15]). A combination of membership and equivalence queries in a polynomial-time algorithm for learning regular languages was presented by Angluin [2]. In her approach, regular languages are represented by state-minimal deterministic finite automata and the algorithm is polynomial in the number of states of a minimal automaton for the target language and the length of the longest counterexample received as response from an equivalence query. It was shown in [4,24] that there exists no polynomial-time learning algorithm that would identify the regular languages either from equivalence queries or from membership queries alone.

Here we will concentrate on learning from positive and negative samples. In particular, we will apply one of the well-known algorithms for learning regular languages from positive and negative examples called RPNI —- *Regular Positive and Negative Inference* by Oncina and Garcia [27].

Informally, the RPNI algorithm starts by separating the set $S$ of all sample words (also called a training set) into a set of positive samples $S^+$, which are words belonging to the target language, and a set of negative samples $S^-$, which are words not belonging to the target language. At first, the algorithm builds a prefix tree automaton from all positive samples. The automaton has states which correspond to all prefixes for all positive samples in $S^+$ and it accepts exactly all words from $S^+$ (i.e. a finite language). Then the algorithm traverses the states of the current automaton and tries to merge pairs of states. If, after merging a pair, no negative sample from $S^-$ is accepted by the resulting quotient automaton, the merged automaton becomes the current automaton, otherwise the automaton is not changed and the algorithm proceeds trying to merge following pairs in a predetermined order.

The running time of the algorithm is $O(\|S^+\| + \|S^-\| \cdot \|S^+\|^2)$, where $\|X\|$, for a set of words $X$, denotes the number of states of a prefix tree automaton accepting all strings from $X$. The algorithm is guaranteed to produce a finite state automaton consistent with the set of samples $S$. This means that the resulting automaton accepts all words from $S^+$ and rejects all words from $S^-$.

## 4 Learning Picture Description Languages

What follows is an informal and intuitive insight into formal definitions and results from [8].

Using the definitions from Section 2, any picture consisting of a connected set of unit-length lines in the square grid of the Cartesian plane together with its starting and ending point can be represented by a string over the alphabet $\Pi = \{l, r, u, d\}$. Then, the picture $pic(w)$, for $w \in \Pi^*$, can be viewed as an equivalence class as many pictures can be translated into one another and because a pen may traverse the same line multiple times. Thus, there are infinitely many words that represent a given nonempty picture. For instance, both pictures $pic(uuddr)$ and $pic(rluduuddr)$ are identical as they define the same set of unit lines and they have the same starting and end points.

We say that a regular language $L$ is a *description language* of a picture $q$ if $\forall w \in L : pic(w) = q$. Let $\mathcal{B}_q$ consist of all description languages of $q$. The regular language $des(q)$ of all descriptions of a picture $q$ belongs to $\mathcal{B}_q$.

For any pictures $q_1$, $q_2$, their concatenation $q_1q_2$ is the picture obtained by joining the end point of $q_1$ and the starting point of $q_2$. Now we can define the class of *regular picture sets* $\mathcal{P}$. It is the minimal class of picture sets satisfying the following:

1. The empty set is in $\mathcal{P}$.
2. For each picture $q$, $\{q\}$ is in $\mathcal{P}$.
3. If $\mathcal{B}_1$ and $\mathcal{B}_2$ are in $\mathcal{P}$, then also
   (a) $\mathcal{B}_1 \cup \mathcal{B}_2$ is in $\mathcal{P}$,
   (b) $\mathcal{B}_1 \cdot \mathcal{B}_2 = \{q_1q_2 \mid q_1 \in \mathcal{B}_1,\ q_2 \in \mathcal{B}_2\}$ is in $\mathcal{P}$, and
   (c) $\mathcal{B}^* = \bigcup_{i \geq 0} \mathcal{B}^i$ is in $\mathcal{P}$, where $\mathcal{B}^{i+1} = \mathcal{B}^i \cdot \mathcal{B}$.

Brüggemann-Klein et al. [8] considered several grammatical inference problems for regular picture sets. They showed that inferring a description language for a fixed picture $q$ can be as hard as inferring arbitrary regular language. Let

$\mathcal{B}_n$ denote the set of all complete descriptions of pictures of size $n$, i.e. having $n$ unit line segments. Each reasonable learning algorithm for $\mathcal{B}_n$ requires $\Omega(2^n)$ membership queries in the worst case.

The inductive structure of such $\mathcal{P}$ allows [8] to resent them as regular expressions over the alphabet $\Pi = l, r, u, d$. However, some regular expressions over $\Pi$ do denote different regular languages of words but the same picture set. When inferring regular pictures sets, membership queries are not sufficient for exact identification of regular picture sets.

Due to the ambiguity of representation of regular picture sets by regular expressions over $\Pi$, the authors in [8] conclude that the methods they proposed cannot be applied to them except for a very special case of *staircase pictures*, whose regular sets can be denoted by regular expressions over the alphabet $\{r, u\}$.

The results on inferring regular picture sets carry over to drawn symbolic languages of [11]. A drawn picture is another term for describing unit lines drawn on the Cartesian plane. By adding an alphabet symbol to each point of the picture we get an intuitive and simple extension called *drawn symbolic picture*. It can be easily seen that these picture languages can also describe picture languages as per the definition in Section 2. Similarly to picture sets, they can be represented by strings. Several important results are proven for regular drawn picture languages: The membership problem is NP-complete [35] and the equivalence, containment and ambiguity problems are undecidable [17,18].

With respect to grammatical inference of picture languages, the above regular picture sets and drawn symbolic picture languages share one problem: on given input picture (without its starting and ending points), how to obtain its string description. This problem was not studied yet.

## 5 Two-Dimensional Automata for Picture Languages

While for string languages we have hierarchies such as the Chomsky hierarchy [10], picture languages lack a similar concept. Moreover, it seems that the taxonomy of languages accepted by various two-dimensional automata is more complicated and so far there has been no complete description of such.

There are known several models of 2D automata [14,6,16,31,32] that generalize the class of regular languages into two dimensions and whose restriction to a single dimension leads to a regular language.

Naturally, for learning of automata it is necessary for us to be able to do all computations effectively. This is hardly achievable for nondeterministic models like online tessellation automaton [14] or sgraffito automaton [31,33] or two-dimensional limited context restarting automaton (2LCRA, for short) [21]. For all these two-dimensional automata models, the membership problem is NP-complete as it is NP-complete already for the REC class accepted by online tessellation automata. The classes of languages accepted by sgrafitto automata and by 2LCRA coincide and they are proper supersets of REC. In contrast to these automata, for the nondeterministic versions of 2LCRA the membership problem is solvable in polynomial time, if the automaton has a so-called correctness preserving property, i.e. if the input word is from the accepted language, then each rewriting operation leads again to a word of the same language [21]. Unfortunately, to find out whether a given automaton has the correctness preserving property is algorithmically undecidable problem [21].

### 5.1 Learning Two-Dimensional Automata

There have been many experiments with learning of automata. Typically those were supervised learning methods. For one-dimensional automata there exist interesting algorithms such as L* [2], RPNI [26], EDSM [22], Biermann-Feldman algorithm [7] and more, whose analogies or generalizations into two dimensions are not known.

Each learning algorithm for two-dimensional automata requires to test acceptance by such automata. Hence, deterministic automata with polynomial complexity of computations are necessary. Nondeterminism of computations of two-dimensional automata can be reduced by limiting the freedom of movements of a head within the picture. E.g., a scanning strategy prescribes the sequence of movements of a head within the picture that visits each field of the picture in a tiling automaton [5] and restarting tiling automaton [32].

Determinism can be achieved by other means as well. It is possible to limit the freedom of application of rules – for example, the first plausible rule is used [29] or the automaton is allowed to (deterministically) choose the movement through the picture by itself based on what it finds in the picture [25,28,29].

The author of [20] proposed an alternative approach to simulation via sgraffito automaton with employing a solver for Constraint Satisfaction Problem: A list of all feasible reductions is generated by the algorithm for every position of the picture. Then sets of locally compatible reductions are chosen until either the generated reductions are exhausted, in which case the input picture is rejected, or a set is found that forms a valid computation of 2LCRA, resulting in acceptance of the input picture. This method is not very efficient and in order to obtain results in a reasonable time, several enhancements and dedicated heuristics and data structures need to be used.

## 6 A New Representation for Picture Languages

Below we propose a new method for learning picture languages based on a new type of representations of picture languages. The new representation of a picture language consists of two parts: a function $R : \Sigma^{*,*} \to (\Sigma \cup \{\#\})^*$ which rewrites any two-dimensional picture over $\Sigma$ into a string over $\Sigma \cup \{\#\}$ and a (string) language $L \subseteq (\Sigma \cup \{\#\})^*$. Then $(R, L)$-picture language is the set of all pictures $P \in \Sigma^{*,*}$ such that $R(P)$ is in $L$. We say that a string language $L^{\mathrm{S}}$ $R$-represents a picture language $L^{\mathrm{P}}$, if for each picture $P$ it holds that $P \in L^{\mathrm{P}}$ if and only if $R(P) \in L^{\mathrm{S}}$.

There are possible many different methods for rewriting a picture into a string. Let us discuss some of them.

Any rectangular picture $P \in \Sigma^{*,*}$ can be represented by storing all symbols (colors of pixels) into a string in the row-by-row fashion. Such simple representation does not contain the information about dimensions of the picture. Therefore, we can add delimiter # for marking ends of rows. E.g., the picture of dimension 3-by-4 containing only letters $a$ will be represented by the string $aaaa\#aaaa\#aaaa$ and the language $L_1^{\mathrm{P}}$ of all nonempty pictures filled with $b$'s will be represented by the (string) language $L_1^{\mathrm{S}} = \{(b^n\#)^{m-1}b^n \mid m, n > 0\}$. Evidently, the language $L_1^{\mathrm{S}}$ is context-sensitive but not context-free. Not all words over $\{b, \#\}$ represent pictures. We would prefer a representation which allows to represent simple picture languages (like $L_1^{\mathrm{P}}$) by simple (e.g. low in the Chomsky hierarchy) string languages. Fortunately, the sample picture language $L_1^{\mathrm{P}}$ can be also represented by the string language defined by the regular expression $(b^+\#)^*b^+$.

The above representation does not allow any simple representation for many elementary picture languages, like the picture language $L_{\mathrm{vl}}^{\mathrm{P}}$ of all pictures containing a column of $b$'s and $a$'s elsewhere. Such language could be represented using a function $R$ which rewrites the picture into a string column-by-column. However instead of allowing different paths during rewriting pictures into strings, we propose to use a window of size 3-by-3. For a picture $P$, the function $R_{\mathrm{w}}$ will scan the extended picture $\widehat{P}$ with the window row-by-row and store the contents of the window (also row-by-row) into a string. E.g., the picture of dimensions 2-by-3 containing $a$'s in the first and last column and $b$'s in the middle column will rewritten into the string

$$\#\#\#\#ab\#ab|\#\#\#abaaba|\#\#\#ba\#ba\#|\#ab\#ab\#\#\#|abaaba\#\#\#|ba\#ba\#\#\#\#$$

(the vertical lines are not present in the string, we have added them in order to separate contents of the window from different positions). Intuitively, such representation can preserve the information about neighboring rows and columns of the picture.

## 6.1 Experimental Results

Our ultimate goal is to learn target picture language $L$ from given sets $S^+$ and $S^-$ of positive and negative sample pictures. That is, we know that $S^+ \subseteq L$ and $S^- \subseteq \Sigma^{*,*} \setminus L$. First, we apply the picture-to-string function $R_{\mathrm{rw}}$ to all known sample pictures. Then we use $R_{\mathrm{rw}}(S^+)$ and $R_{\mathrm{rw}}(S^-)$ as positive and negative samples to infer an $R_{\mathrm{rw}}$-representation of $L$ using RPNI algorithm.

To experimentally verify the proposed learning protocol, we have generated random positive and negative examples of four picture languages. All our sample picture languages are over the binary alphabet $\{a, b\}$:

$L_1$ is the set of all rectangular pictures (not necessarily square) filled with $b$'s and containing a diagonal of $a$'s either from the top-left corner or from the top-right corner. The diagonal leads till the border of the picture.

$L_2$ is the set of all pictures which contain several rows filled with $a$'s followed by rows of $b'$ till the bottom of the picture.

$L_3$ is the set of all pictures of dimensions at least $(3, 3)$ filled with $b$'s except border filled with $a$'s.

$L_4$ is the set of all pictures with regular chessboard pattern of $a$'s and $b$'s. The top-left corner of such picture can contain $a$ or $b$, but the whole picture must have the chessboard pattern.

Sample pictures from each of the sample languages are shown in Table 1.

**Table 1.** Sample pictures from the picture languages $L_1$, $L_2$, $L_3$ and $L_4$.

| | | | | | |
|---|---|---|---|---|---|
| # | # | # | # | # | # |
| # | a | b | b | b | # |
| # | b | a | b | b | # |
| # | b | b | a | b | # |
| # | b | b | b | a | # |
| # | # | # | # | # | # |

| | | | | | |
|---|---|---|---|---|---|
| # | # | # | # | # | # |
| # | a | a | a | a | # |
| # | a | a | a | a | # |
| # | a | a | a | a | # |
| # | b | b | b | b | # |
| # | # | # | # | # | # |

| | | | | | |
|---|---|---|---|---|---|
| # | # | # | # | # | # |
| # | a | a | a | a | # |
| # | a | b | b | a | # |
| # | a | b | b | a | # |
| # | a | a | a | a | # |
| # | # | # | # | # | # |

| | | | | | |
|---|---|---|---|---|---|
| # | # | # | # | # | # |
| # | a | b | a | b | # |
| # | b | a | b | a | # |
| # | a | b | a | b | # |
| # | b | a | b | a | # |
| # | # | # | # | # | # |

In our experiments we have generated different training and testing sets of positive and negative sample pictures for each of the sample languages. Our sets of sample pictures contained pictures of dimensions between 3 and 8 such that each training set contained approximately the same number of positive and negative samples.

By applying the function $R_{\text{rw}}$ with window size 3-by-3 we have rewritten the sample pictures into sample strings. Then for each set $S$ of sample strings, we have learned a deterministic finite state automaton consistent with $S$. For that we have used an RPNI implementation in MATLAB from [1]. Afterwards, we tested the resulting automaton on an independent test set of pictures (rewritten into strings by the function $R_{\text{rw}}$). The learned finite automata correctly recognize the training set of pictures, however they do not accept/reject correctly all pictures from the test sets.

Below in Table 2 we report accuracy – the ratio of correctly accepted/rejected sample pictures within a test set of the same size as the training set used to infer the corresponding automaton. We have noted also running times for the algorithm RPNI for different sizes of training sets. All experiments were performed on a 64 bit Windows PC with Intel Core i5-4460 processor running at 3.2 GHz.

The absolute value for the running time is not so important as we believe that the running times can be substantially reduced by re-implementing the RPNI algorithm in C or C++ and by using a more recent model of CPU. The interesting information is that the running time of the RPNI algorithm varies for different picture languages considerably. In particular, the language $L_2$ seems to be "easy" for RPNI compared to other sample languages.

## 6.2 Possible Extensions

The above representation does not allow any simple representation for many elementary picture languages, like the picture language $L_{\text{sq}}^{\text{P}}$ of all square pictures containing $b$'s. Let us extend the above representation. Instead of collecting all symbols (colors of pixels) of a picture in a fixed order, we will allow "to draw" the picture by traversing in an arbitrary order encoded in the string representation together with symbols of the fields of the picture. For that let $\Pi_0 = \{l, r, u, d\}$ be a fixed alphabet encoding directions of movements to the left, right, up and down. Then a word $w = w_1 w_2 \dots w_n$, where

**Table 2.** Accuracy and time (in seconds) for learning sample languages for different sizes of training sets.

| Lang. | 100 examples | | 200 examples | | 400 examples | | 800 examples | | 1600 examples | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Acc | Time | Acc | Time | Acc | Time | Acc | Time | Acc | Time |
| $L_1$ | 0.84 | 35.67 | 0.84 | 244.22 | 0.91 | 996.12 | 0.90 | 1316.04 | 0.97 | 964.99 |
| $L_2$ | 1.00 | 1.68 | 1.00 | 3.72 | 1.00 | 7.67 | 1.00 | 11.63 | 0.99 | 317.12 |
| $L_3$ | 0.86 | 31.62 | 0.88 | 134.52 | 0.89 | 396.32 | 0.95 | 531.85 | 0.95 | 2883.87 |
| $L_4$ | 0.83 | 62.31 | 0.83 | 185.98 | 0.91 | 413.62 | 0.91 | 1532.75 | 0.91 | 2175.82 |

$w_i \in \Pi^*$, where $\Pi = \Pi_0 \cup \Sigma \cup \{\#\}$ for $i = 1, \ldots n$, can be interpreted as follows. The interpretation starts with $i = 1$ and the row-column position $(x_1, y_1) = (1, 1)$ at the top left corner of the picture. If $w_1 = \#$, then $n = 1$ and the word encodes the empty picture. Otherwise, if $w_i \in \Sigma \cup \{\#\}$, then the field at position $(x_i, y_i)$ is "paint" by the symbol $w_i$ and the position for the next step does not change: $(x_{i+1}, y_{i+1}) = (x_i, y_i)$. Let $\eta : \Pi \to \mathbb{Z} \times \mathbb{Z}$ be the mapping $\eta(l) = (0, -1)$, $\eta(r) = (0, 1)$, $\eta(u) = (-1, 0)$, $\eta(d) = (0, 1)$, and $\eta(x) = (0, 0)$ for all $x \in \Sigma \cup \{\#\}$. If $w_i \in \Pi$, then the position changes to $(x_{i+1}, y_{i+1}) = (x_i, y_i) + \eta(w_i)$. Interpreting a word $w$ in this way does not ensure, that each position of the drawn picture is assigned a symbol from $\Sigma$. Therefore, we will assume that the fields of the picture which were not visited or visited without assigning a symbol from $\Sigma$ have a fixed "background" color $\alpha \in \Sigma$. We say that a picture $P \in \Sigma^{*,*}$ of dimension $(m, n)$ was painted by the word $w \in \Pi^*$ if

- starting in the top left field of the rectangular picture $P_0 \in \{\alpha\}^{m,n}$, where $\alpha$ is the fixed background symbol, during interpretation of $w$ on the bordered picture $\widehat{P_0}$, we do not leave the tape where $\widehat{P_0}$ is stored and
- the resulting picture is $P$.

For example, if $a$ is the background symbol, by interpreting $(brd)^{n-1}br\#dl\#$ we paint the square picture of dimension $(n, n)$, for $n \geq 1$, with diagonal marked by the symbols $b$ and with symbols $a$ elsewhere. Then the string language $(rd)^*u\#dl\#$ represents all "empty" square pictures. On the other hand, single word *brbdblb* represents all pictures of size $(m, n)$ where $m, n \geq 2$ consisting of symbols $a$ except of small 2-by-2 square containing four symbols $b$ located at the top left corner of the picture.

## 7 Conclusions

We have introduced a new representation of picture languages which differs from known ones. It is based on a function that rewrites given picture into a string and a string language that is used to decide whether the picture belongs to the picture language or not. Evidently, the complexity of recognizing whether the picture is from the picture language depends mainly on the complexity of the membership problem for the string language.

We experimented with the rewriting function $R_{\mathrm{rw}}$ which rewrites the picture by storing the contents of a scanning window of size 3-by-3 when scanning given picture row by row. The obtained results show that using this function and a classical grammatical inference algorithm RPNI, which is capable of inferring regular languages, we can infer some simple picture languages.

Nevertheless, this paper is only the first step in applying the new representation of picture languages for inferring picture languages. Further theoretical study is needed to establish the power of such representations with respect to known classes of picture languages. By combining different functions for rewriting pictures into strings and different classes of string languages (or automata to recognize them) we can obtain a rich set of picture language classes.

The results of our experiments encourage more thorough study employing other known methods for inferring string languages, like evidence driven state merging algorithms [9].

**Acknowledgement.** This research was supported by the Charles University Grant Agency (GAUK) project no. 1198519.

## References

1. Akram, H.I., De La Higuera, C., Xiao, H., Eckert, C.: Grammatical inference algorithms in matlab. In: International Colloquium on Grammatical Inference. pp. 262–266. Springer (2010)
2. Angluin, D.: Learning regular sets from queries and counterexamples. Information and computation 75(2), 87–106 (1987)
3. Angluin, D.: Queries and concept learning. Machine learning 2(4), 319–342 (1988)
4. Angluin, D.: Negative results for equivalence queries. Machine Learning 5(2), 121–150 (1990)
5. Anselmo, M., Giammarresi, D., Madonia, M.: Tiling automaton: A computational model for recognizable two-dimensional languages. In: International Conference on Implementation and Application of Automata. pp. 290–302. Springer (2007)
6. Anselmo, M., Giammarresi, D., Madonia, M.: A computational model for tiling recognizable two-dimensional languages. Theoretical Computer Science 410(37), 3520–3529 (2009)
7. Biermann, A.W., Feldman, J.A.: On the synthesis of finite-state machines from samples of their behavior. IEEE Trans. Comput. 21(6), 592–597 (Jun 1972)
8. Brüggemann-Klein, A., Fischer, P., Ottmann, T.: Learning picture sets from examples. In: Results and Trends in Theoretical Computer Science, pp. 34–43. Springer (1994)
9. Bugalho, M., Oliveira, A.L.: Inference of regular languages using state merging algorithms with search. Pattern Recognition 38(9), 1457–1467 (2005)
10. Chomsky, N.: Three models for the description of language. IRE Transactions on information theory 2(3), 113–124 (1956)
11. Costagliola, G., Deufemia, V., Ferrucci, F., Gravino, C.: On regular drawn symbolic picture languages. Information and Computation 187(2), 209–245 (2003)
12. Freeman, H.: On the encoding of arbitrary geometric configurations. IRE Transactions on Electronic Computers EC-10(2), 260–268 (June 1961)
13. Giammarresi, D., Restivo, A.: Recognizable picture languages. International Journal of Pattern Recognition and Artificial Intelligence 6(02n03), 241–256 (1992)
14. Giammarresi, D., Restivo, A.: Two-Dimensional Languages, pp. 215–267. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)

15. De la Higuera, C.: Grammatical inference: learning automata and grammars. Cambridge University Press (2010)
16. Inoue, K., Nakamura, A.: Two-dimensional multipass on-line tessellation acceptors. Information and Control 41(3), 305–323 (1979)
17. Kim, C.: Complexity and decidability for restricted classes of picture languages. Theoretical Computer Science 73(3), 295–311 (1990)
18. Kim, C., Sudborough, I.H.: The membership and equivalence problems for picture languages. Theoretical Computer Science 52(3), 177–191 (1987)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
20. Krtek, L.: Learning picture languages using restarting automata. master thesis, Charles University, Faculty of Mathematics and Physics (2014)
21. Krtek, L., Mráz, F.: Two-dimensional limited context restarting automata. Fundamenta Informaticae 148(3-4), 309–340 (2016)
22. Lang, K.J., Pearlmutter, B.A., Price, R.A.: Results of the abbadingo one dfa learning competition and a new evidence-driven state merging algorithm. In: International Colloquium on Grammatical Inference. pp. 1–12. Springer (1998)
23. Maurer, H.A., Rozenberg, G., Welzl, E.: Using string languages to describe picture languages. Information and Control 54(3), 155–185 (1982)
24. Moore, E.F.: Gedanken-experiments on sequential machines. Automata studies 34, 129–153 (1956)
25. Mráz, F., Otto, F.: Ordered restarting automata for picture languages. In: International Conference on Current Trends in Theory and Practice of Informatics. pp. 431–442. Springer (2014)
26. Oncina, J., Garcia, P.: Identifying regular languages in polynomial time. In: Advances in structural and syntactic pattern recognition, pp. 99–108. World Scientific (1992)
27. Oncina, J., García, P.: Inferring regular languages in polynomial updated time. In: Pattern recognition and image analysis: selected papers from the IVth Spanish Symposium. pp. 49–61. World Scientific (1992)
28. Otto, F., Mráz, F.: Extended two-way ordered restarting automata for picture languages. In: International Conference on Language and Automata Theory and Applications. pp. 541–552. Springer (2014)
29. Otto, F., Mráz, F.: Deterministic ordered restarting automata for picture languages. Acta Informatica 52(7-8), 593–623 (2015)
30. Pradella, M., Crespi Reghizzi, S.: A sat-based parser and completer for pictures specified by tiling. Pattern Recogn. 41(2), 555–566 (Feb 2008)
31. Průša, D., Mráz, F.: Two-dimensional sgraffito automata. In: International Conference on Developments in Language Theory. pp. 251–262. Springer (2012)
32. Průša, D., Mráz, F.: Restarting tiling automata. International Journal of Foundations of Computer Science 24(06), 863–878 (2013)
33. Průša, D., Mráz, F., Otto, F.: Two-dimensional sgraffito automata. RAIRO-Theoretical Informatics and Applications 48(5), 505–539 (2014)
34. Sakakibara, Y.: Learning context-free grammars from structural data in polynomial time. Theoretical Computer Science 76(2-3), 223–242 (1990)
35. Sudborough, I.H., Welzl, E.: Complexity and decidability for chain code picture languages. Theoretical Computer Science 36, 173–202 (1985)