Binary Image Operator Design based on Stacked Generalization

Nina S. T. Hirata
Department of Computer Science
Institute of Mathematics and Statistics
University of São Paulo
Rua do Matão, 1010 — 05508-090 São Paulo, Brazil
nina@ime.usp.br

Abstract

Stacked generalization refers to any learning schema that consists of multiple levels of training. Level zero classifiers are those that depend solely on input data while classifiers at other levels may use the output of lower levels as the input. Stacked generalization can be used to address the difficulties related to the design of image operators defined on large windows. This paper describes a simple stacked generalization schema for the design of binary image operators and presents several application examples that show its effectiveness as a training schema.

1. Introduction

Translation-invariant and locally defined binary image operators are characterized by Boolean functions. The number of variables in the Boolean function is given by the size of the window that establishes the locality property of the operator.

In order to design operators of this family, one only needs to choose a Boolean function. If we consider images to be processed as random realizations of a random set $\mathbf{S}$ and their respective ideal images as random realizations of a random set $\mathbf{I}$, then the goal of image operator design is to obtain a mapping Ψ such that $\Psi(\mathbf{S})$ is statistically as close as possible to $\mathbf{I}$. The closeness between $\Psi(\mathbf{S})$ and $\mathbf{I}$ can be expressed in terms of some measure that involves the joint distribution of $\mathbf{S} \times \mathbf{I}$.

In practice, however, the distribution that governs $\mathbf{S} \times \mathbf{I}$ is not known. The design of image operators is usually based on learning techniques where probabilities are estimated from samples of input-output training data [1, 2]. For a fixed amount of training data, if the window considered is large, the operators have poor statistical precision and the results are not good. In general, we can not simply reduce the window size to address the problem of precision because by doing so we limit the ability of the operator to discriminate relatively large shapes. Moreover, the time spent as well as the amount of computer memory required to train operators that depend on large windows is usually very large (training may take several days).

Some approaches to deal with the difficulties related to designing image operators based on large windows have been suggested. Iterative operators [3] consider sequential composition of several operators as a way of successively approximating the output to the ideal output. Two-stage learning [6] divides the window into two sub-windows, the internal and external sub-windows, learns a relaxed mapping with respect to the external sub-window, and then combines the output of this mapping with the input corresponding to the internal sub-window in an optimal fashion.

In this paper we propose the use of stacked generalization schema to address the problem of designing large window binary image operators. Stacked generalization [7] refers to any learning schema that consists of multiple levels of training. Level zero classifiers are those that depend solely on input data while classifiers at other levels may use the output of lower levels as the input. The iterative and the two-stage approaches cited above can be understood as particular cases of the stacked generalization schema.

Following this Introduction, this paper is organized as follows. Section 2 introduces notation and concepts related to binary image operator design. Section 3 briefly recalls stacked generalization and then shows how it can be settled in the context of binary image operator design. A simple schema considers that all classifiers target one same output pixel. A more general schema considers that different classifiers at the intermediary levels of the schema may target distinct output pixels. Section 4 describes experimental results of designing binary image operators based on stacked generalization schema for texture segmentation, character recognition and text segmentation. Finally, Section 5 points out some concluding remarks and future research issues re-

lated to this subject.

2. Binary Image Operator Design

Binary images can be seen as subsets of Z^2. The set corresponding to a binary image $f : Z^2 \to \{0,1\}$ is $S_f = \{z \in Z^2 : f(z) = 1\}$ and, conversely, the binary image f_S corresponding to a set $S \subseteq Z^2$ is given by $f_S(z) = 1$ if $z \in S$ and $f_S(z) = 0$ if $z \notin S$.

Let $\mathcal{P}(Z^2)$ (the set all subsets of Z^2) denote the set of all binary images defined on Z^2, and let $W \subseteq Z^2$ be an n-point window. The patterns (subsets) observed through W by sliding it on an image S can be seen as elements of $\mathcal{P}(W)$. Thus, functions of the form $\psi : \mathcal{P}(W) \to \{0,1\}$ can be used to define binary image operators in the following way. For any binary image S and $z \in Z^2$,

$$z \in \Psi(S) \Longleftrightarrow \psi(\, S_{-z} \cap W \,) = 1 \tag{1}$$

where $S_{-z} \cap W$ denotes the pattern observed at location z, translated to the origin. We say that ψ is the characteristic function of Ψ and that Ψ is a W-operator.

If we assign a binary variable to each point of the window, then function $\psi : \mathcal{P}(W) \to \{0,1\}$ can be understood as a Boolean function, that is, as a mapping from $\{0,1\}^n$ to $\{0,1\}$. For instance, if we consider a 3-point window and three variables x_1, x_2, and x_3, assigned respectively to each point of the window, the Boolean function $\psi(x_1, x_2, x_3) = x_1\, x_2 + x_1\, x_3 + x_2\, x_3$ defines a binary median operator.

In the context of this paper, a relevant fact is that Boolean functions characterize a class of binary image operators and, therefore, the problem of designing these operators reduces to the problem of designing Boolean functions.

The problem of designing W-operators can be stated as follows. We consider that images to be processed and their respective ideal result images are modeled respectively as random sets $\mathbf{S}$ and $\mathbf{I}$. We would like to design an image operator Ψ so that $\Psi(\mathbf{S})$ is statistically as close as possible to $\mathbf{I}$. We suppose that $\mathbf{S}$ and $\mathbf{I}$ are jointly stationary, so that patterns observed through a window W on the input images and the corresponding value in the output image are considered realizations of a joint distribution $P(\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ is a random set whose realizations are in $\mathcal{P}(W)$ and $\mathbf{y}$ is a binary random variable with realizations in $\{0,1\}$. A usual closeness measure between $\Psi(\mathbf{S})$ and $\mathbf{I}$ is the mean absolute error (MAE). Owing to stationarity, we can define the MAE of an operator Ψ characterized by ψ, with respect to distribution $P(\mathbf{X}, \mathbf{y})$, as

$$MAE\langle\Psi\rangle = E\Big[\, \big|\, \psi(\mathbf{X}) - \mathbf{y} \,\big|\, \Big] \tag{2}$$

where $E[\cdot]$ denotes the expected value of its argument. Thus, the minimum MAE (MMAE) operator is the one characterized by function

$$\psi(X) = \begin{cases} 1, & \text{if } P(X,1) > P(X,0), \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

If the joint distribution $P(\mathbf{X}, \mathbf{y})$ is known, then MMAE operator can be obtained from Eq. 3. However, in practice, these probabilities are not known. They are usually estimated from samples of input-output pairs of images. Patterns collected on the input images through window W and their respective output values are used to estimate the distribution $P(\mathbf{X}, \mathbf{y})$. A training procedure can, therefore, be viewed as consisting of the following steps:

1. Scan each input-output pair of training set and estimate $P(\mathbf{x}, y)$. This gives an estimate $\hat{P}(X, y)$ of $P(X, y)$ for some pairs (X, y), $X \in \mathcal{P}(W)$ and $y \in \{0,1\}$.

2. For each observed pattern X, estimate MAE optimal classification $\psi(X)$ as defined in Equation 3, using $\hat{P}(X, y)$ instead of $P(X, y)$.

3. To complete the definition of function ψ, apply some learning algorithm on the partially defined function ψ.

Step 2 above determines the classification of the patterns present in the images of the training set. In this sense, we may think that after step 2 is accomplished, we have a partially defined function ψ. In step 3, two additional issues must be addressed: (1) how to completely define ψ, and (2) how to represent function ψ. These issues are automatically addressed by most of the learning algorithms. Examples of such algorithms are decision trees, Boolean function minimization, artificial neural networks, etc. All these algorithms generate an implicit representation of a function (classifier) and are able to classify any pattern in $\mathcal{P}(W)$.

A serious issue in classifier design is the curse of dimensionality [4]. In the context of image operator design it means that the use of large window does not necessarily yield a good result. It has been observed that, for a fixed training data set, if operators for increasing size windows are trained, the MAE tends to initially decrease, then reaches a minimum point, and then starts to increase as the window gets larger and larger. The error of operators with small window is due to their inability to discriminate shapes larger than the dimension of the window, and the error of operators with larger windows is due to the statistical imprecision (the amount of data is not enough to obtain a good estimate of the joint probabilities).

To overcome this problem, a number of approaches have been proposed. Among them, some rely on the representation of functions as composition of other functions depending on a smaller number of variables [6, 3].

3. Stacked Generalization

In classifier design, a set of training examples is used to learn a classifier. Generalization is the ability of a classifier to correctly classify unseen examples. The term stacked generalization was introduced by Wolpert in 1992 [7] and is related to the problem of how to deal with multiple classifiers. Stacked generalization refers to any learning schema that consists of multiple levels of training. Level zero classifiers are those that depend solely on input data while classifiers at other levels may use the output of lower levels as the input.

In this paper we consider stacked generalization restricted to binary classifiers of the form $f : \{0,1\}^n \rightarrow \{0,1\}$. The basic idea behind stacked generalization is the following. Suppose we have a set of classifiers $f_1, f_2, \ldots, f_p$ trained over a training set A. If these are asked to guess the output corresponding to an example $\mathbf{x}$ in another set B, their guess will be $f_1(\mathbf{x})$, $f_2(\mathbf{x})$, ..., and $f_p(\mathbf{x})$ respectively. Since none of them was trained on B, it may be possible that none of them makes the correct guess y. However, a new information becomes available: we know that when the guesses of $f_1, f_2, \ldots, f_p$ for $\mathbf{x}$ are respectively $f_1(\mathbf{x})$, $f_2(\mathbf{x})$, ..., and $f_p(\mathbf{x})$, the correct guess is y. Therefore, the elements of the type $\big((f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_p(\mathbf{x})),\, y\big)$ can be used to train a new classifier f that incorporates information from $f_1, f_2, \ldots, f_p$.

The classifiers $f_1, f_2, \ldots, f_p$ are called level-0 classifiers and f is called a level-1 classifier. The number of levels can be increased by combining classifiers of lower levels. This training schema is denominated stacked generalization. It is also referred to as multi-stage classification. Notice that several variants of these schema are possible. For instance, the level 0 classifiers can be classifiers trained with different learning algorithms, or with training sets of different sizes, or with fewer than n inputs and so on. Analogously, classifiers at any other level may have as the input not only classifiers of the level immediately below it but from any of the levels below it, including some of the n initial inputs. Moreover, zero and intermediary level classifiers may assign examples to meta-classes. Only the final level classifier need to assign a valid class to the examples.

3.1. Stacked Generalization in Binary Image Operator Design

Stacked generalization can be used as a way to overcome window size problem in the image operator design. Instead designing one function $\psi(x_1, x_2, \ldots, x_n)$ with respect to a large window W of size n, one can split W into smaller sub-windows and design functions with respect to these smaller sub-windows. These functions would form the level-0 classifiers. Another function would be used to combine these functions, forming the level-1 classifier.

This paper considers only two-level stacked generalization schemas. Let (S, I) be a pair of training images. Training examples are the patterns observed on S through a window W. If X is a pattern located at position z in S, then the value y of I at position z is the target class. The simplest stacked schema is to consider p sub-windows $W_1, W_2, \ldots, W_p \subseteq W$ with $W_1 \cup W_2 \cup \cdots \cup W_p = W$ and p respective level zero classifiers to be denoted ψ_1^0, ψ_2^0, ..., ψ_p^0, and one level 1 classifier that integrates the results of these p level zero classifiers. If we denote the level 1 classifier as ψ^1, we have

$$\psi^1 = \psi^1_{1,2,\ldots,p}\big(\psi_1^0, \psi_2^0, \ldots, \psi_p^0\big) \tag{4}$$

where ψ^1 is a mapping from $\mathcal{P}(W)$ to $\{0,1\}$, ψ_i^0 is a mapping from $\mathcal{P}(W_i)$ to $\{0,1\}$ ($i = 1, 2, \ldots, p$), and $\psi^1_{1,2,\ldots,p}$ is a mapping from $\{0,1\}^p$, to $\{0,1\}$. To keep notation consistent, we denote the operator designed on W as ψ^0.

We may have all level zero classifiers targeting the same output y as well as distinct relative locations in the output image. If they target distinct locations in the output image, then those locations can be thought as defining another window W' with p points. Each level zero function ψ_i^0 would estimate an output y_i assigned to a point in W'. Level 1 function would be of type $\psi : \mathcal{P}(W') \rightarrow \{0,1\}$. An example of this schema, with three level zero classifiers targeting distinct locations in the output image, is illustrated in Fig. 1. Notice that sub-windows may or not overlap.

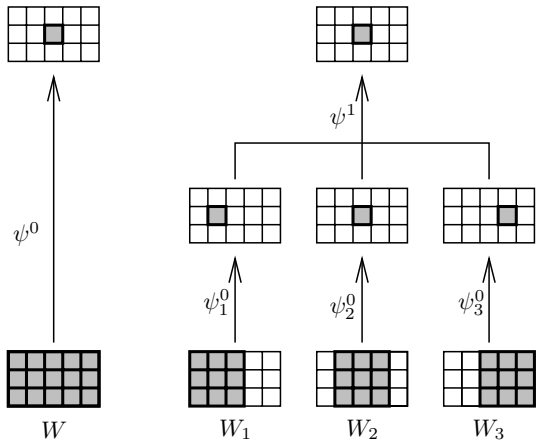


Figure 1. Two-level stacked generalization schema.

In the iterative design technique [3], pairs of training images (S, I) are used to design the first stage operator Ψ_1, pairs of the type $(\Psi_1(S), I)$ are used to train the second stage operator Ψ_2, and successively so that pairs of

the type $(\Psi_{k-1}(\ldots(\Psi_1(\Psi_0(S)))\ldots), I)$ are used to train the k-th stage operator Ψ_k. The proposed schema includes iterative design as a particular case. In the two level schema, two-stage iterative design corresponds to considering a same ψ_i^0 for all $i = 1, 2, \ldots, p$. It is noteworthy to mention that the proposed schema relates to the multilevel decomposition of Boolean functions. In particular, Curtis/Ashenhurst [5] decomposition deals with a particular case where sub-windows do not overlap. Also, by considering adequate sub-samplings of the large window, it is possible to have operators dealing with multi-resolution data.

4. Experimental Results

This section reports some experimental results of the application of stacked generalization schema in binary image operator design. Following notations introduced in previous sections, we denote by ψ^0 the operator designed directly on the entire window, by ψ_i^0 the level 0 operator designed over sub-window W_i, and by ψ^1 the level 1 operator that integrates information from level 0 operators. Our objective is to compare ψ^1 (the operator designed according to the proposed stacked schema) with ψ^0 (the operator designed without stacking). In these experiments, all level zero classifiers target the same pixel in the output image.

4.1. Texture segmentation

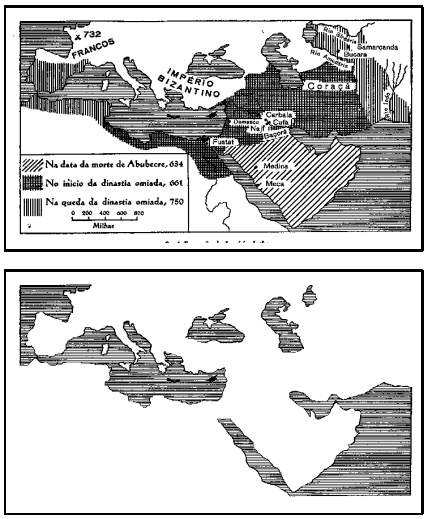


Figure 2. Example of training data for texture segmentation.

Texture is a key feature in many image analysis procedures such as surface classification, image retrieval and satellite or aerial image analysis. Figure 2 shows a pair of input-output images used to train operators for texture segmentation.

Figure 3 shows the windows used in this experiment. Crossed window point indicates the relative location of the target pixel in the output image.

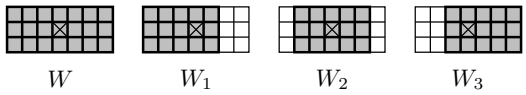
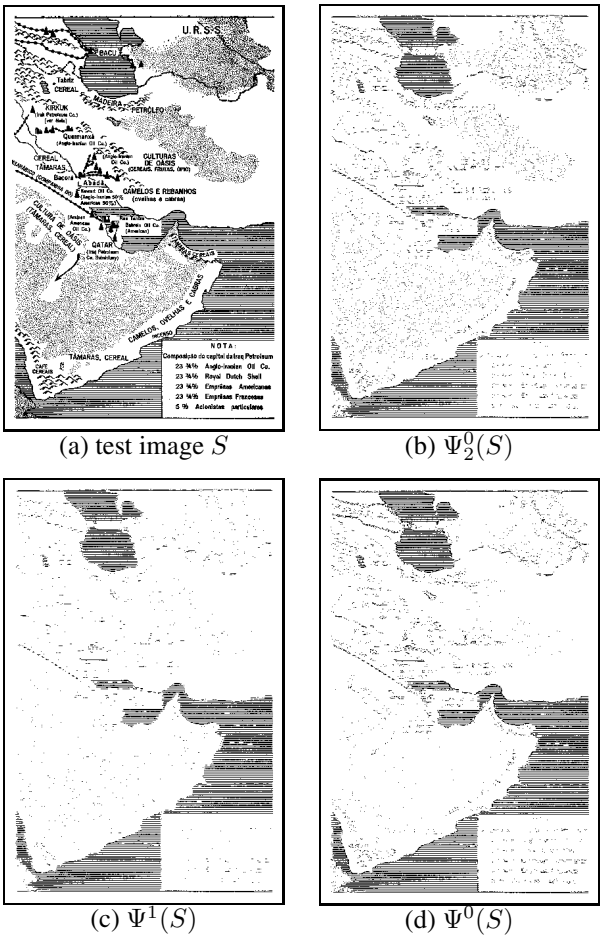


Figure 3. Windows used in experiment for texture segmentation.

Figure 4 shows a test image S, the result $\Psi_2^0(S)$ of one of the level 0 operators, the result $\Psi^1(S) = \Psi^1_{1,2,3}(\Psi_1^0(S), \Psi_2^0(S), \Psi_3^0(S))$ of level 1 operator, and the result of the operator $\Psi^0(S)$ designed directly on W. As we can see, $\Psi^0(S)$ is better than $\Psi_2^0(S)$ but is no better than $\Psi^1(S)$.



(a) test image S (b) $\Psi_2^0(S)$

(c) $\Psi^1(S)$ (d) $\Psi^0(S)$

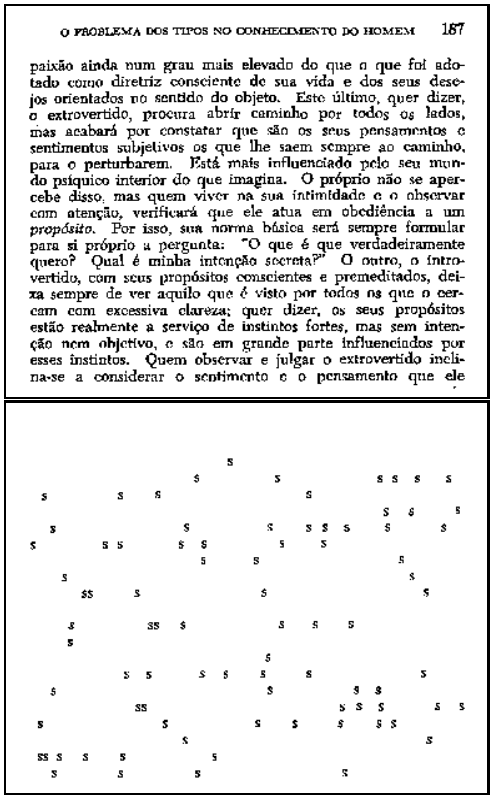
Figure 4. Result of experiment for texture segmentation.

Level 0 operators ψ_1^0, ψ_2^0, and ψ_3^0 for windows W_1, W_2, and W_3, respectively, were trained with images number 1 and 2. Level 1 operator $\psi_{1,2,3}^1$ was trained with image number 4. An operator ψ^0 was trained for window $W = W_1 \cup W_2 \cup W_3$ with images number 1, 2, and 4. Table 1 summarizes the result. Stacked schema operator ψ^1 outperforms ψ^0 both in precision (MAE) and training time.

Operator	window	training time (m:s)	Error (#pixels) image 3	image 5
ψ_1^0	W_1	00:36	6792	9277
ψ_2^0	W_2	00:35	6909	9236
ψ_3^0	W_3	00:35	6903	9318
ψ^1		00:00	5944	6242
ψ^0	W	23:21	6727	7276

Table 1. Summary of experiment for texture segmentation.

4.2. Character recognition


O PROBLEMA DOS TIPOS NO CONHECIMENTO DO HOMEM 187

paixão ainda num grau mais elevado do que o que foi adotado como diretriz consciente de sua vida e dos seus desejos orientados no sentido do objeto. Este último, quer dizer, o extrovertido, procura abrir caminho por todos os lados, mas acabará por constatar que são os seus pensamentos e sentimentos subjetivos os que lhe saem sempre ao caminho, para o perturbarem. Está mais influenciado pelo seu mundo psíquico interior do que imagina. O próprio não se apercebe disso, mas quem viver na sua intimidade e o observar com atenção, verificará que ele atua em obediência a um *propósito*. Por isso, sua norma básica será sempre formular para si próprio a pergunta: "O que é que verdadeiramente quero? Qual é minha intenção secreta?" O outro, o introvertido, com seus propósitos conscientes e premeditados, deixa sempre de ver aquilo que é visto por todos os que o cercam com excessiva clareza; quer dizer, os seus propósitos estão realmente a serviço de instintos fortes, mas sem intenção nem objetivo, e são em grande parte influenciados por esses instintos. Quem observar e julgar o extrovertido inclina-se a considerar o sentimento e o pensamento que ele

Figure 5. Example of training data for character recognition.

Figure 5 shows a pair of images used to design operators for recognition of character s (lowercase letter s). These images were obtained by scanning pages of a book written in Portuguese with resolution of 200dpi and then reducing their size to the half of the originals.

Experiment 1: Five level 0 operators were trained over each of the windows shown in Fig. 6, using 3 pairs of training images. Level 1 operator were trained using 2 additional pairs of training images. Table 2 shows the details of these operators.

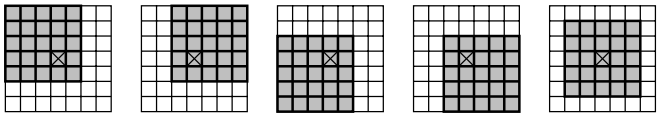


Figure 6. Windows of Experiment 1 for character recognition.

Operator	window	training time (m:s)	Error (#pixels) im. 1	im. 4	im. 10
ψ_1^0	W_1	00:13	559	585	672
ψ_2^0	W_2	00:11	511	536	577
ψ_3^0	W_3	00:11	437	529	613
ψ_4^0	W_4	00:09	563	614	673
ψ_5^0	W_5	00:12	300	320	333
ψ^1		00:00	128	156	217
ψ^0	W	02:38	202	205	270

Table 2. Summary of Experiment 1 for character recognition.

Experiment 2: Experiment 1 were repeated for the windows shown in Fig. 7.

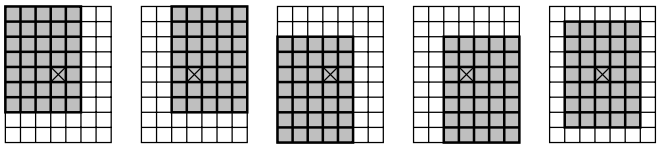
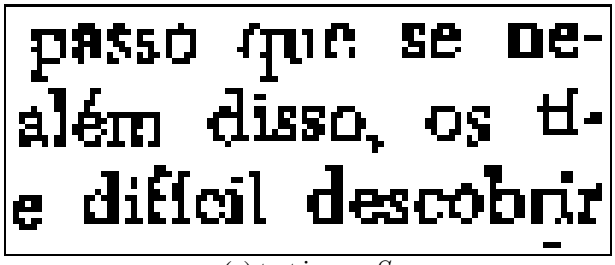
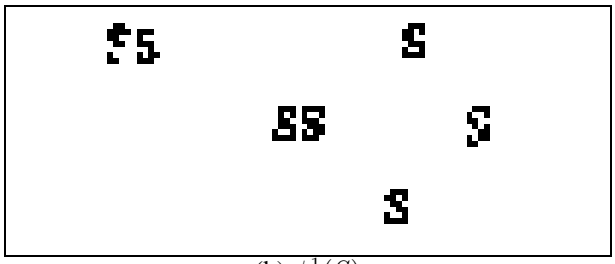


Figure 7. Windows of Experiment 2 for character recognition.

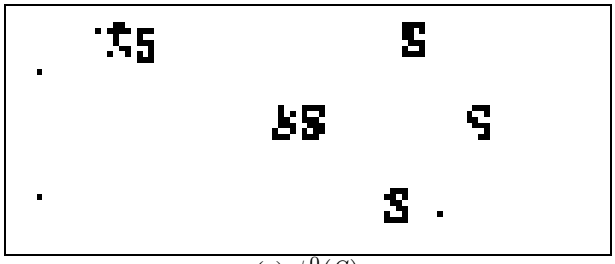
Figure 8 shows a subregion of a test image and the results of ψ^1 and ψ^0.



(a) test image S



(b) $\psi^1(S)$



(c) $\psi^0(S)$

Figure 8. Result of Experiment 2 for character recognition.

As shown in Table 3, the stacked operator outperforms the non-stacked one consistently but the difference is small (visually they are almost imperceptible) and we could say that in terms of recognition power they are equivalent. However we should note that the overall training time of the stacked operator is less than the training time of ψ^0.

In experiment 1, the overall training time of the stacked operator is 56 seconds while the training time of the direct operator is 2 minutes and 38 seconds. In experiment 2, the overall training time of the stacked operator is 2 minutes and 11 seconds while the training time of the direct operator is 4 minutes and 25 seconds. Moreover, error of ψ^0 with 9×7 window (Experiment 2) is larger than the error of ψ^0 with 7×7 window (Experiment 1). This suggests that operators designed on larger windows using the same amount of training data will probably present poorer performance. On the other hand, it seems that there is still room to improve stacked schema operators by, for instance, using 7×7 window level 0 operators.

Operator	window	training time (m:s)	Error (#pixels)		
			im. 1	im. 4	im. 10
ψ_1^0	W_1	00:28	356	368	417
ψ_2^0	W_2	00:28	323	342	399
ψ_3^0	W_3	00:24	253	284	402
ψ_4^0	W_4	00:27	364	359	458
ψ_5^0	W_5	00:24	223	226	319
ψ^1		00:00	99	129	162
ψ^0	W	04:25	206	216	304

Table 3. Summary of Experiment 2 for character recognition.

4.3. Text segmentation

In document processing and analysis, prior to application of an OCR for character recognition, an important task is to segment text regions from the document.

Figure 9 shows a 15×15 window and a 5×5 sub-window within it. Other eight 5×5 sub-windows can be defined within window 15×15 by varying the way points are subsampled in each of the 3×3 squares.

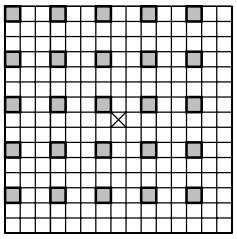


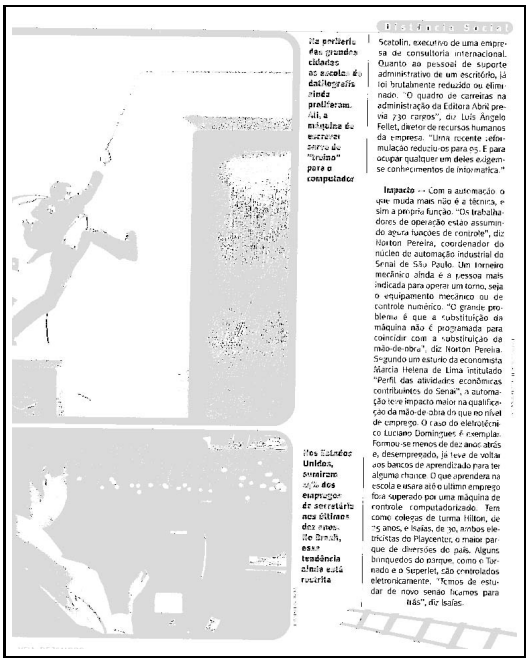
Figure 9. One of the windows of experiment for text segmentation.

Nine level 0 operators were trained with respect to the nine 5×5 sub-windows using 1 pair of training images. A second pair of images was used to train the level 1 operator ψ^1. No operator were trained on the 15×15 window because it would demand a very large training time. We compare the stacked operator to the operator trained with respect to a 5×5 window with the origin at its center.

Figure 10 shows an example of training image pair used to design an operator for text segmentation.

Figure 10. Example of training data for text segmentation.

(a) Stacked generalization approach



(b) 5×5 window

Figure 11. Result of experiment for text segmentation.

Figure 11 shows a test image processed by a single 5×5 window operator and processed by the one designed by the stacked generalization schema and Table 4 compiles the results obtained with these operators.

Operator	window	training time (h:m:s)	Error (#pixels) image 8	image 20
ψ_1^0	W_1	1:05:47	5475	7578
ψ_2^0	W_2	1:31:12	5277	8026
ψ_3^0	W_3	1:11:20	5546	7458
ψ_4^0	W_4	1:45:30	5695	7499
ψ_5^0	W_5	1:27:01	5360	7405
ψ_6^0	W_6	1:22:40	5534	7376
ψ_7^0	W_7	2:36:02	5075	6850
ψ_8^0	W_8	1:52:47	5439	7296
ψ_9^0	W_9	2:11:11	5205	6921
ψ^1		0	2153	3863
$\psi_{5\times5}$	5×5	1:33:19	6843	7143

Table 4. Summary of experiment for text segmentation.

5. Concluding Remarks

Stacked generalization was proposed as a training schema to design large window binary image operators. It includes as particular cases the iterative and the multi-stage design techniques. It also has the power to deal with images in a multi-resolution fashion. Experimental results with two level stacked generalization schema, with level 0 classifiers designed over sub-windows of the large window, show that the proposed method consistently outperforms the single level classifier designed over the large window both in MAE and in training time. The reasons that explain these results were not addressed in this paper and they are certainly an interesting issue to be investigated. Investigations in this direction may profit from some known results in the field of multiple classifier systems.

Some immediate improvements can be obtained by combining this schema with the iterative design technique. At each iteration, this schema would be used to obtain a good classifier with respect to the window considered in the iteration.

The experimental results presented in this paper refer to the schema in which all classifiers target the same single pixel in the output image. Therefore, output is decided solely on basis of the input data. The application of the general stacked generalization schema, that allows intermediate level functions to target different pixels in the output image, may be useful to incorporate geometrical information of the output image in the design process. We believe that this would be helpful, for instance in a problem of edge detection, to correct broken edges.

The proposed approach can be easily extended to gray-level operator design. A challenging issue to be addressed in future works is the automatic training of operators based on this schema, by automatically selecting parameters such as the window, the number of levels, the number of classifiers in each level and its sub-windows.

Acknowledgements

The author is partially supported by CNPq (Brasil), grant 300238/01-0.

References

[1] J. Barrera, E. R. Dougherty, and N. S. Tomita. Automatic Programming of Binary Morphological Machines by Design of Statistically Optimal Operators in the Context of Computational Learning Theory. *Electronic Imaging*, 6(1):54–67, January 1997.

[2] J. Barrera, R. Terada, R. Hirata Jr, and N. S. T. Hirata. Automatic Programming of Morphological Machines by PAC Learning. *Fundamenta Informaticae*, 41(1-2):229–258, January 2000.

[3] N. S. T. Hirata, E. R. Dougherty, and J. Barrera. Iterative Design of Morphological Binary Image Operators. *Optical Engineering*, 39(12):3106–3123, December 2000.

[4] A. K. Jain, R. P. Duin, and J. Mao. Statistical Pattern Recognition: A Review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), January 2000.

[5] J. P. Roth and R. M. Karp. Minimization over boolean graphs. *IBM Journal of Research and Development*, 6(2):227–238, 1962.

[6] O. V. Sarca, E. Dougherty, and J. T. Astola. Two-stage Binary Filters. *Electronic Imaging*, 8(3):219–232, July 1999.

[7] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.