Precipitates Segmentation from Scanning Electron Microscope Images Through Machine Learning Techniques

João P. Papa[1], Clayton R. Pereira[1], Victor H. C. de Albuquerque[2], Cleiton C. Silva[3], Alexandre X. Falcão[4], and João Manuel R. S. Tavares[5]

[1]Dep. of Computing, UNESP - Univ Estadual Paulista, Bauru, Brazil
{papa,clayton}@fc.unesp.br
[2]Center of Technological Sciences, University of Fortaleza, Fortaleza, Brazil
victor.albuquerque@fe.up.pt
[3]Dep. of Materials and Metallurgical Engineering, Federal University of Ceará, Brazil
cleitonufc@yahoo.com.br
[4]Institute of Computing, State University of Campinas, Campinas, Brazil
afalcao@ic.unicamp.br
[5]Faculty of Engineering, University of Porto, Porto, Portugal
tavares@fe.up.pt

Abstract. The presence of precipitates in metallic materials affects its durability, resistance and mechanical properties. Hence, its automatic identification by image processing and machine learning techniques may lead to reliable and efficient assessments on the materials. In this paper, we introduce four widely used supervised pattern recognition techniques to accomplish metallic precipitates segmentation in scanning electron microscope images from dissimilar welding on a Hastelloy C-276 alloy: Support Vector Machines, Optimum-Path Forest, Self Organizing Maps and a Bayesian classifier. Experimental results demonstrated that all classifiers achieved similar recognition rates with good results validated by an expert in metallographic image analysis.

Key words: Support Vector Machines, Optimum-Path Forest, Scanning Electron Microscope, Metallic Precipitates Segmentation, Hastelloy C-276

1 Introduction

Nickel based alloys are an important class of metallic materials especially employed under severe operational conditions, mainly because of its high temperature strength and resistance to corrosion/oxidation. In this context, Hastelloy C276 alloy has been notably used as protective coating against corrosion on inner surface in equipments from petroleum and petrochemical industries due to the high contents of chromium, molybdenum, and tungsten.

Among many manufacturing process used to deposit the coating, the arc welding process is one of the most important. During the solidification of the

liquid metal in the weld pool, a phenomenon of microsegregation from solid dendrite to liquid interdendritic is observed for some elements as molybdenum and tungsten [14].

In the final stage of solidification, the liquid enriched in Mo (molybdenum) and W (tungsten) originates a new phase, known as topologically closed packed (TCP) phase [14], which is detrimental to mechanical properties due to its hard and brittle nature [1]. Besides that, the resistance to corrosion of these type of alloys can be decreased by the precipitation of the Mo-rich TCP phase. In addition, the formation of TCP phases are responsible for the weld metal hot cracks in Hastelloy C-276 [5]. In order to avoid or minimize these deleterious phases, it has a consensus about choosing the welding parameters in a properly manner.

Nonetheless, to verify the effect of welding parameters on the formation of TCP phases it is often necessary to accurately identify the amount of precipitates, which are responsible to decrease the mechanical properties of metallic materials. Thus, it is very important to have an effective tool to identify and further quantity the material precipitates and microstrutures, in order to assess the quality of metallic materials as a whole.

Albuquerque et al. [2] have addressed this problem using image processing techniques together with machine learning ones in order to speed up the process and to make it less prone to errors inherent to human inspection. In that work, it was presented and evaluated computational solutions for segmentation and quantification of different types of cast iron microstructures from optical microscopy images based on Artificial Neuronal Networks with Multilayer Perceptron (ANN-MLP)and Self-Organizing Maps (SOM), which were compared against a commercial system. As far as we know, only Albuquerque et al. [4] tackled the problem of material precipitates segmentation using images obtained from scanning electron microscope (SEM).

Hence, we propose in this work to apply machine learning techniques that have not been applied to this context up to date, such as: Optimum-Path Forest (OPF), two different implementations of Support Vector Machines (SVMs), SOM and a Bayesian classifier. The remainder of the paper is organized as follows. As the OPF classifier was recently introduced, we dedicated a Section to introduce its fundaments and learning algorithm (Section 2). Section 3 addresses the methodology and the used dataset. The experimental results are addressed in Section 4 and the conclusions are stated in Section 5.

2 Pattern Recognition by Optimum-Path Forest

The Optimum-Path Forest is a framework to assist the development of pattern recognition techniques based on optimum-path forest [12]. An OPF-based classifier models the problem of pattern recognition as a graph partition in a feature space induced by the dataset. Each sample is represented by a set of features and a distance function measures their dissimilarity in the feature space. The training samples are then interpreted as the nodes of a graph, whose arcs are

defined by a given adjacency relation and weighted by the distance function. It is expected that samples from a same class/cluster are connected by a path of nearby samples. Therefore, the degree of connectedness for any given path is measured by a connectivity (path-value) function, which exploits the distances along the path.

In supervised learning, the true label of the training samples is known and so it is exploited to identify key samples (prototypes) in each class. Optimum paths are computed from the prototypes to each training sample, such that each prototype becomes root of an optimum-path tree composed by its most strongly connected samples. The labels of these samples are assumed to be the same of their root. In unsupervised learning, each cluster is represented by an optimum-path tree rooted at a single prototype but we do not know the class label of the training samples. Therefore, we expect that each cluster contains only samples of a same class and some other information about the application is needed to complete classification.

The basic idea is then to specify an adjacency relation and a path-value function, compute prototypes and reduce the problem into an optimum-path forest computation in the underlying graph. The training forest becomes a classifier which can assign to any new sample the label of its most strongly connected root. Essentially, this methodology extends a previous approach, called *Image Foresting Transform* [8], for the design of image processing operators from the image domain to the feature space.

Papa et al. [11] presented a first method for supervised classification using a complete graph (implicit representation) and the maximum arc weight along a path as connectivity function. The prototypes were chosen as samples that share an arc between distinct classes in a minimum spanning tree of the training set [7].

Another supervised learning method was proposed in [10]. In this case, the arcs connect k-nearest neighbors (k-nn) in the feature space. The distances between adjacent nodes are used to estimate a probability density value of each node and optimum paths are computed from the maxima of this probability density function (pdf). For large datasets, we usually use a smaller training set and a much larger evaluation set to learn the most representative samples from the classification errors in the evaluation set. This considerably improves classification accuracy of new samples. This strategy was assessed with k-nn graphs in [13]. The accuracy results can be better than using similar strategy with complete graph [11] for some situations, but the latter is still preferred because it is faster and does not require the optimization of the parameter k.

An unsupervised version of OPF was presented by Rocha et al. [15], which is quite similar to the supervised one with k-nn graph. The main difference rely on the estimation of the best k value: in this case, as we do not have information about labels, the k value chosen is the one that minimizes the minimum cut over the whole graph. In this paper, we adopted the OPF with complete graph, since that this version is the most used. For sake of simplicity, any further refer-

ence to OPF will mean this version. The next sections will details the training, classification and the learning with pruning procedures for OPF.

2.1 Training

In large datasets, the number of labeled samples for training is usually large. Therefore, a first strategy to make a classifier more efficient is the use of two labeled and disjoint sets, Z_1 and Z_2, $|Z_1| \ll |Z_2|$, being the first the actual training set and the second an evaluation set. The purpose of the evaluation set is to improve the quality of the samples in the training set, without increasing its size, by replacing classification errors in Z_2 by non-prototype samples of Z_1 [11]. After this learning process, the classifier is ready to be tested on any unseen dataset Z_3. For validation purpose, this process must also be repeated several times, with different random and disjoint sets Z_1, Z_2, and Z_3, in order to obtain the average accuracy results.

Let (Z_1, A) be a complete graph whose nodes are the samples in Z_1 and any pair of samples defines an arc in $A = Z_1 \times Z_1$. The arcs do not need to be stored and so the graph representation is implicit. A path is a sequence of distinct samples $\pi_t = \langle s_1, s_2, \ldots, s_{k-1}, t\rangle$ with terminus t, where $(s_i, s_{i+1}) \in A$ for $1 \leq i \leq k-1$. A path is said *trivial* if $\pi_t = \langle t\rangle$. We assign to each path π_t a cost $f(\pi_t)$ given by a path-value function f. A path π_t is considered optimum if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t with the same terminus t. We also denote by $\pi_s \cdot \langle s, t\rangle$ the concatenation of a path π_s and arc (s, t).

Training essentially consists of finding an optimum-path forest in (Z_1, A), which is rooted in a special set $S \subset Z_1$ of prototypes. As proposed in [11], the set S is represented by samples that share arcs between distinct classes in a minimum-spanning tree (MST) of (Z_1, A) [7]. For path-value function $f_{\max}$, these prototypes (roots of the forest) tend to minimize the classification errors in Z_1, when their labels are propagated to the nodes of their trees:

$$f_{max}(\langle s\rangle) = \begin{cases} 0 & \text{if } s \in S \\ +\infty & \text{otherwise,} \end{cases}$$
$$f_{max}(\pi_s \cdot \langle s, t\rangle) = \max\{f_{max}(\pi_s), d(s, t)\}, \quad (1)$$

such that $f_{max}(\pi_s)$ computes the maximum distance between adjacent samples in a non-trivial path π_s.

The training algorithm for the OPF classifier [11] assigns one optimum path $P_1^*(s)$ from S to every sample $s \in Z_1$, forming an optimum path forest P_1 (a function with no cycles which assigns to each $s \in Z_1 \backslash S$ its predecessor $P_1(s)$ in $P_1^*(s)$ or a marker *nil* when $s \in S$). Let $R_1(s) \in S$ be the root of $P_1^*(s)$ (which can be reached from $P_1(s)$), the OPF algorithm computes for each $s \in Z_1$, the minimum cost $C_1(s)$ of $P_1^*(s)$, the class label $L_1(s) = \lambda(R_1(s))$, and the predecessor $P_1(s)$. *Algorithm 1* implements this training procedure.

Algorithm 1 – Training Algorithm

INPUT: A λ-labeled training set Z_1 and the pair (v, d) for feature vector and distance computations.
OUTPUT: Optimum-path forest P_1, cost map C_1, label map L_1, and ordered set Z_1'.
AUXILIARY: Priority queue Q, set S of prototypes, and cost variable *cst*.

1. *Set* $Z_1' \leftarrow \emptyset$ *and compute by MST the prototype set* $S \subset Z_1$.
2. *For each* $s \in Z_1 \backslash S$, *set* $C_1(s) \leftarrow +\infty$.
3. *For each* $s \in S$, *do*
4. $\quad$ $C_1(s) \leftarrow 0$, $P_1(s) \leftarrow nil$, $L_1(s) \leftarrow \lambda(s)$, *and insert* s *in* Q.
5. *While* Q *is not empty, do*
6. $\quad$ *Remove from* Q *a sample* s *such that* $C_1(s)$ *is minimum.*
7. $\quad$ *Insert* s *in* Z_1'.
8. $\quad$ *For each* $t \in Z_1$ *such that* $t \neq s$ *and* $C_1(t) > C_1(s)$, *do*
9. $\quad\quad$ *Compute* $cst \leftarrow \max\{C_1(s), d(s,t)\}$.
10. $\quad\quad$ *If* $cst < C_1(t)$, *then*
11. $\quad\quad\quad$ *If* $C_1(t) \neq +\infty$, *then remove* t *from* Q.
12. $\quad\quad\quad$ $P_1(t) \leftarrow s$, $L_1(t) \leftarrow L_1(s)$, $C_1(t) \leftarrow cst$.
13. $\quad\quad\quad$ *Insert* t *in* Q.
14. *Return a classifier* $[P_1, C_1, L_1, Z_1']$.

2.2 Classification

In [11], the classification of each new sample $t \in Z_2$ (or Z_3) is done based on the distance $d(s,t)$ between t and each training node $s \in Z_1$ and on the evaluation of the following equation:

$$C_2(t) = \min\{\max\{C_1(s), d(s,t)\}\}, \ \forall s \in Z_1. \tag{2}$$

Let $s^* \in Z_1'$ be the node s that satisfies Equation (2). It essentially considers all possible paths π_s from S in (Z_1, A) extended to t by an arc (s,t), finds the optimum path $P_1^*(s^*) \cdot \langle s^*, t \rangle$, and label t with the class $\lambda(R_1(s^*))$ of its most strongly connected prototype $R_1(s^*) \in S$ (i.e., $L_2(t) \leftarrow L_1(s^*) = \lambda(R_1(s^*))$).

Note that Z_1 can be replaced by Z_1' in Equation (2) and its evaluation can halt when $\max\{C_1(s), d(s,t)\} < C_1(s')$ for a node s' whose position in Z_1' succeeds the position of s. This avoids to visit all nodes in Z_1' in many cases and the efficiency gain increases with the time complexity of $d(s,t)$. *Algorithm 2* implements this scheme for classification procedure.

Algorithm 2 – OPF CLASSIFICATION

INPUT: Classifier $[P_1, C_1, L_1, Z_1']$, evaluation set Z_2 (or test set Z_3), and the pair (v, d) for feature vector and distance computations.
OUTPUT: Label L_2 and predecessor P_2 maps defined for Z_2.
AUXILIARY: Cost variables *tmp* and *mincost*.

1. *For each* $t \in Z_2$, *do*
2. $\quad$ $i \leftarrow 1$, $mincost \leftarrow max\{C_1(k_i), d(k_i, t)\}$.
3. $\quad$ $L_2(t) \leftarrow L_1(k_i)$ *and* $P_2(t) \leftarrow k_i$.

4. *While* $i < |Z_1'|$ *and* $mincost > C_1(k_{i+1})$, *do*
5. *Compute* $tmp \leftarrow \max\{C_1(k_{i+1}, d(k_{i+1}, t)\}$.
6. *If* $tmp < mincost$, *then*
7. $mincost \leftarrow tmp$.
8. $L_2(t) \leftarrow L(k_{i+1})$ *and* $P_2(t) \leftarrow k_{i+1}$.
9. $i \leftarrow i + 1$.
10. *Return* $[L_2, P_2]$.

In *Algorithm 2*, the main loop (Lines $1-9$) performs classification of all nodes in Z_2. The inner loop (Lines $4-9$) visits each node $k_{i+1} \in Z_1'$, $i = 1, 2, \ldots, |Z_1'|$ until an optimum path $\pi_{k_{i+1}} \cdot \langle k_{i+1}, t\rangle$ be found. In the worst scenario, it visits all nodes in Z_1' (Line 4). Line 5 evaluates $f_{max}(\pi_{k_{i+1}} \cdot \langle k_{i+1}, t\rangle)$ and Lines $7-8$ updates cost, label and predecessor of t whenever $\pi_{k_{i+1}} \cdot \langle k_{i+1}, t\rangle$ is better than the current path π_t (Line 6).

2.3 Pruning Irrelevant Patterns

Large datasets usually present redundancy, so at least in theory it should be possible to estimate a reduced training set with the most relevant patterns for classification. The use of a training set Z_1 and an evaluation set Z_2 has allowed us to learn relevant samples for Z_1 from the classification errors in Z_2, by swapping misclassified samples of Z_2 and non-prototype samples of Z_1 during a few iterations [11]. In this learning strategy, Z_1 remains with the same size and the classifier instance with the highest accuracy is selected to be tested in the unseen set Z_3. In this section, we use this learning procedure (as described in *Algorithm 3*) within a method (*Algorithm 4*) to reduce the training set size by identifying and eliminating irrelevant samples from Z_1.

Algorithm 3 – OPF Learning Algorithm

Input:	A λ-labeled training and evaluating sets Z_1 and Z_2, respectively, number T of iterations, and the pair (v, d) for feature vector and distance computations.
Output:	Optimum-path forest P_1, cost map C_1, label map L_1, and ordered set Z_1'.
Auxiliary:	Arrays FP and FN of sizes c for false positives and false negatives, set S of prototypes, and list LM of misclassified samples.

1. *Set* $MaxAcc \leftarrow -1$.
2. *For each iteration* $I = 1, 2, \ldots, T$, *do*
3. $LM \leftarrow \emptyset$ *and compute the set* $S \subset Z_1$ *of prototypes.*
4. $[P_1, C_1, L_1, Z_1'] \leftarrow$ *Algorithm 1* $(Z_1, S, (v, d))$.
5. *For each class* $i = 1, 2, \ldots, c$, *do*
6. $FP(i) \leftarrow 0$ *and* $FN(i) \leftarrow 0$.
7. $[L_2, P_2] \leftarrow$ *Algorithm 2* $(Z_1', Z_2, (v, d))$
8. *For each sample* $t \in Z_2$, *do*
9. *If* $L_2(t) \neq \lambda(t)$, *then*
10. $FP(L_2(t)) \leftarrow FP(L_2(t)) + 1$.

11. $FN(\lambda(t)) \leftarrow FN(\lambda(t)) + 1$.
12. $LM \leftarrow LM \cup t$.
13. *Compute accuracy Acc according to [11].*
14. *If* $Acc > MaxAcc$ *then save the current instance* $[P_1, C_1, L_1, Z_1']$
15. *of the classifier and set* $MaxAcc \leftarrow Acc$.
16. *While* $LM \neq \emptyset$
17. $LM \leftarrow LM \backslash t$.
18. *Replace t by a non-prototype sample, randomly selected from* Z_1.
19. *Return the classifier instance* $[P_1, C_1, L_1, Z_1']$ *with the highest accuracy in* Z_2.

The efficacy of *Algorithm 3* increases with the size of Z_1, because more non-prototype samples can be swapped by misclassified samples of Z_2. However, for sake of efficiency, we need to choose some reasonable maximum size for Z_1. After learning the best training samples for Z_1, we may also mark paths in P_1 used to classify samples in Z_2 and define their nodes as *relevant samples* in a set $\mathcal{R}$. The "irrelevant" training samples in $Z_1 \backslash \mathcal{R}$ can then be moved to Z_2. *Algorithm 4* applies this idea repetitively, while the loss in accuracy on Z_2 with respect to the highest accuracy obtained by *Algorithm 3* (using the initial training set size) is less or equal to a maximum value $MLoss$ specified by the user.

Algorithm 4 – Learning-with-Pruning Algorithm

Input: Training and evaluation sets, Z_1 and Z_2, labeled by λ, the pair (v, d) for feature vector and distance computations, maximum loss $MLoss$ in accuracy on Z_2, and number T of iterations.
Output: EOPF classifier $[P_1, C_1, L_1, Z_1']$ with reduced training set.
Auxiliary: Set $\mathcal{R}$ of relevant samples, and variables *Acc* and *tmp*.

1. $[P_1, C_1, L_1, Z_1'] \leftarrow$ *Algorithm 3* $(Z_1, Z_2, T, (v, d))$.
2. $[L_2, P_2] \leftarrow$ *Algorithm 2*$(Z_1', Z_2, (v, d))$ *and store accuracy in Acc.*
3. $tmp \leftarrow Acc$ *and* $\mathcal{R} \leftarrow \emptyset$.
4. *While* $|Acc - tmp| \leq MLoss$ *and* $\mathcal{R} \neq Z_1$ *do*
5. $\mathcal{R} \leftarrow \emptyset$.
6. *For each sample* $t \in Z_2$, *do*
7. $s \leftarrow P_2(t) \in Z_1$.
8. *While* $s \neq nil$, *do*
9. $\mathcal{R} \leftarrow \mathcal{R} \cup s$.
10. $s \leftarrow P_1(s)$.
11. *Move samples from* $Z_1 \backslash \mathcal{R}$ *to* Z_2.
12. $[P_1, C_1, L_1, Z_1'] \leftarrow$ *Algorithm 3* $(Z_1, Z_2, T, (v, d))$.
13. $[L_2, P_2] \leftarrow$ *Algorithm 2* $(Z_1', Z_2, (v, d))$ *and store accuracy in tmp.*
14. *Return* $[P_1, C_1, L_1, Z_1']$.

In *Algorithm 4*, Lines $1 - 3$ compute learning and classification using the highest accuracy classifier obtained for an initial training set size. Its accuracy is stored in *Acc* and used as reference value *Acc* in order to stop the pruning process, when the loss in accuracy is greater than an user-specified value $MLoss$ or all training samples are considered relevant. The main loop in Lines $4 - 13$

essentially marks the relevant samples in Z_1 by following backwards the optimum paths used for classification (Lines $5-10$), moves irrelevant samples to Z_2, and repeats learning and classification from a reduced training set until it reaches the above stopping criterion.

3 Methodology

The material used to evaluate classifiers was a dissimilar metal weld of Hastelloy C276 alloy on a C-Mn steel substrate. The testing samples were extracted from the welded plates and subsequently went through a metallographic processing, which consists in sanding, polishing and electrolytic etching. All samples were etched using 10% chromic acid, and a 2V tension applied during 15 seconds.

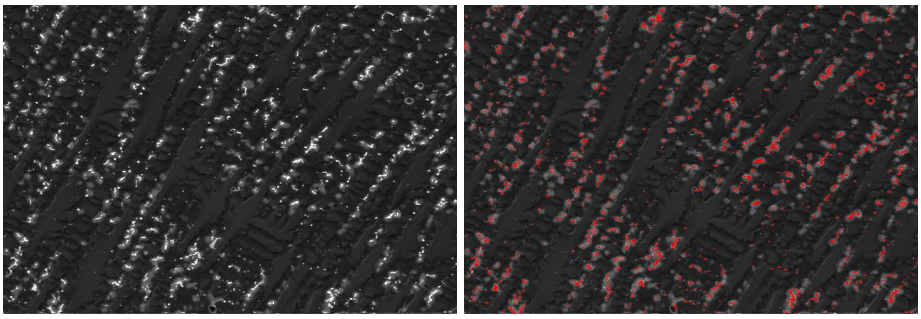
The SEM images were obtained in secondary electron (SE) mode, which presents an adequate contrast between the precipitates and the matrix, due to the enrichment of the precipitates by elements with higher atomic weight such as Mo and W. After the imaging acquisition process, the images were submitted to the analysis of the machine learning solutions under evaluation.

Regarding machine learning techniques, we used here five implementations: Self Organizing Maps (SOM), Optimum-Path Forest (OPF), SVM without kernel mapping (SVM-nokernel), SVM with RBF (Radial Basis Function) as kernel mapping (SVM-RBF) and a Bayesian classifier. For SOM, we used our own implementation with a 5×5 neuronal lattice and 10 iterations for learning. The OPF implementation we used was the one from LibOPF [12], which is a free library of optimum-path forest-based classifiers. For OPF learning algorithm, we used the pruning procedure described in Section 2.3. Regarding SVM-nokernel we adopted LibLINEAR [9] with parameters optimized by cross-validation, and for SVM-RBF we used SVMTorch [6]. Finally, for Bayesian classifier we used our implementation.

As we are working with supervised pattern recognition techniques, it is necessary to have labeled data for the learning process. Thus, we asked for an expert in metallographic image analysis to label an entire image in two classes: foreground (precipitates) and background (matrix). Figure 1 displays the image used to train the classifiers (a) and its respective labeled image (b), in which the red pixels mean the precipitates. In this work, each pixel is considered as a sample to build the dataset, and the feature vector used as input is composed by the gray value of each pixel.

4 Experimental Results

In this section, we present the experiments realized in order to asses the robustness of the classifiers, which were conducted in two phases: in the former (Section 4.1) we used 1% for training and the remaining 99% for classification. The samples were obtained through the whole image shown in Figure 1. In the latter round of experiments (Section 4.2), we used the same 1% above to train the classifiers and further to label another image of the dataset. We conducted



(a) (b)

Fig. 1. (a) SEM image used in the first round of experiments (Section 4.1) and (b) after be labeled by an expert.

an extra experiment in Section 4.1 in order to assess the performance of OPF learning with pruning algorithm described in Section 2.3. For that experiment, we divided the above 1% in 10% for training and 90% for the evaluating set. The $MLoss$ variable in *Algorithm 4* was set to 0.3. Notice that all these values were empirically chosen based on our previous experience.

The accuracies are measured by taking into account that the classes may have different sizes in Z_2 (similar definition is applied for Z_3). If there are two classes, for example, with very different sizes and a classifier always assigns the label of the largest class, its accuracy will fall drastically due to the high error rate on the smallest class.

Let $NZ_2(i)$, $i = 1, 2, \ldots, c$, be the number of samples in Z_2 from each class i. We define the errors $e_{i,1}$ and $e_{i,2}$:

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|} \quad \text{and} \quad e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, \; i = 1, \ldots, c, \tag{3}$$

where $FP(i)$ and $FN(i)$ are the false positives and false negatives, respectively. That is, $FP(i)$ is the number of samples from other classes that were classified as being from the class i in Z_2, and $FN(i)$ is the number of samples from the class i that were incorrectly classified as being from other classes in Z_2. The errors $e_{i,1}$ and $e_{i,2}$ are then used to define:

$$E(i) = e_{i,1} + e_{i,2}, \tag{4}$$

where $E(i)$ is the partial sum error of class i. Finally, the accuracy Acc, are defined as:

$$Acc = \frac{2c - \sum_{i=1}^{c} E(i)}{2c} = 1 - \frac{\sum_{i=1}^{c} E(i)}{2c}. \tag{5}$$

4.1 Robustness of Classifiers

Table 1 display the mean results using 1% for training and 99% for classification after 10 rounds with randomly chosen sets. These experiments were performed

using a PC with Intel® Core I5 processor and 4Gb RAM and Linux Ubuntu 10.04 as the operational system.

Table 1. Mean accuracy and mean training and classification times for OPF, SVM-RBF, SVM-noKernel and SOM.

Classifier	Accuracy %	Training time [s]	Classification Time [s]
OPF	89.86±5.08	0.314	0.594
SVM-RBF	90.84±1.71	0.149	1.672
SVM-noKernel	94.56±2.86	9.965	0.140
SOM	88.87±3.21	0.045	0.065
Bayesian	87.47±1.28	0.025	49.89

Although SVM-noKernel achieved the best results, OPF, SVM-RBF, SOM and Bayesian were also similar if we consider the standard deviation. The fastest classifier for training was Bayesian one, and for classification was SOM, and the best trade-off between efficiency and effectiveness was achieved by OPF, which was the second faster classifier and 11.129 times faster than SVM-noKernel if we take into account the whole execution time, i.e., training plus classification.

Concerning the extra experiment, i.e., the one using OPF learning with pruning algorithm, OPF achieved 91.23% of recognition rate and pruned 97.38% of the training set. Now, the OPF testing time decreased to 0.171 seconds, which turn OPF with training set pruning 3.45 faster than traditional OPF for classification. Note that the accuracy also increased, even reducing the training set.

4.2 Automatic Labeling Images

In this second round of experiments, we applied the same 1% training set used in the previous section to train the classifiers for further labeling another SEM image of the dataset, as illustrated in Figure 2a.

It is possible to observe, from visual assessment, that all approaches achieved reasonable results regarding the quality of the segmentation (Figures 2(b)-(e)). Thus, it feasible to make the Hastelloy C-276 alloy automatic characterization in a precisely and efficiently manner, since that the human inspection may be prone to errors due to the subjectivity of this process, as addressed by Albuquerque et al. [3]. Hence, this work may contribute with a comparison among supervised pattern recognition techniques in order to obtain fast and reliable results to this urged and demanded task.

5 Conclusions

In this paper, we addressed the problem of metallic precipitates segmentation in SEM images, which may affect the material durability and resistance. As there are very few works in this context, the present one assumes a significantly

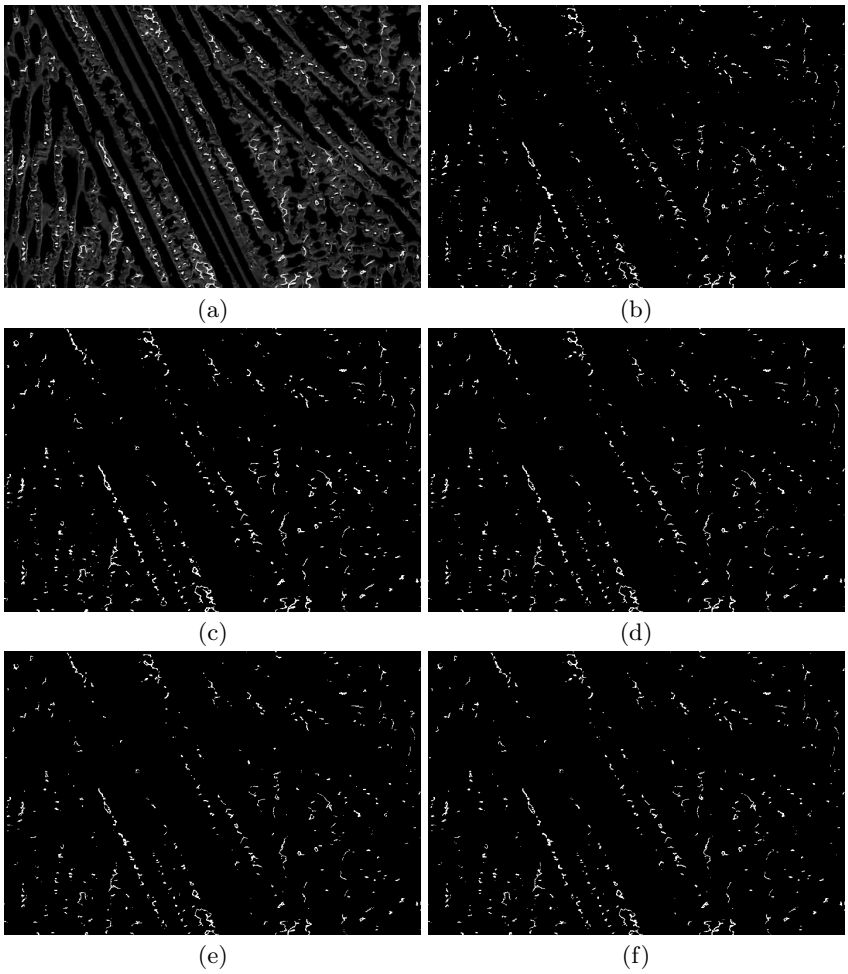


Fig. 2. (a) SEM image used in the second round of experiments and its respectively classified images by (b) OPF, (c) SVM-noKernel, (d) SVM-RBF, (e) SOM and (f) Bayesian.

contribution by performing a comparison among the state-of-the-art supervised pattern recognition to accomplish this task.

We conducted two rounds of experiments: (i) in the former, the accuracy of two different implementations of SVMs, SOM and OPF were compared in terms of effectiveness and efficiency for training and classification, and (ii) in the latter experiment we used the classifiers trained in the previous one to label a another image of the dataset. Regarding accuracy over the classification set, all classifiers were similar if we consider the standard deviation, been SOM the fastest one. Additionally, OPF achieved the best trade-off between effectiveness and efficiency. Finally, in the second round, we attested that all classifiers pro-

duced similar results to label an image that did not belong to the training set. In addiction, these segmentation results were considered feasible by an expert in in metallographic image analysis.

Acknowledgments. The authors thank National Council for Research and Development (CNPq), São Paulo Research Foundation (FAPESP) grants 2009/16206-1 and 2010/02045-3, and Cearense Foundation for the Support of Scientific and Technological Development (FUNCAP) for providing financial support through a DCR grant to UNIFOR for third the author. This work was also partially done in the scope of the project with reference PTDC/EEA-CRO/103320/2008 financially supported by Fundação para a Ciência e a Tecnologia (FCT) in Portugal.

References

1. Akhter, J., Shaikh, M., Ahmad, M., Iqbal, M., Shoaib, K., Ahmad, W.: Effect of aging on the hardness and impact properties of hastelloy c-276. Journal of Materials Science Letters 20(4), 333–335 (2001)
2. de Albuquerque, V.H.C., de Alexandria, A.R., Cortez, P.C., Tavares, J.M.R.S.: Evaluation of multilayer perceptron and self-organizing map neural network topologies applied on microstructure segmentation from metallographic images. NDT & E International 42(7), 644–651 (2009)
3. de Albuquerque, V.H.C., Cortez, P.C., de Alexandria, A.R., Tavares, J.M.R.S.: A new solution for automatic microstructures analysis from images based on a backpropagation artificial neural network. Nondestructive Testing and Evaluation 23(4), 273–283 (2008)
4. de Albuquerque, V.H.C., da Silva, C.C., Menezes, T.I.S., Farias, J., ao Manuel R. S. Tavares, J.: Automatic evaluation of nickel alloy secondary phases from sem images. Microscopy Research and Technique 74(1), 36–46 (2011)
5. Cieslak, M., Headley, T., Romig, A.: The welding metallurgy of hastelloy alloys c-4, c-22, and c-276. Metallurgical and Materials Transactions A 17, 2035–2047 (1986)
6. Collobert, R., Bengio, S.: Svmtorch: support vector machines for large-scale regression problems. The Journal of Machine Learning Research 1, 143–160 (2001)
7. Cormen, T., Leiserson, C., Rivest, R.: Introduction to Algorithms. MIT (1990)
8. Falcão, A.X., Stolfi, J., Lotufo, R.A.: The image foresting transform theory, algorithms, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(1), 19–29 (2004)
9. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. Journal of Machine Learning Research 9, 1871–1874 (2008)
10. Papa, J.P., Falcão, A.X.: A new variant of the optimum-path forest classifier. In: Proceedings of the 4th International Symposium on Advances in Visual Computing. pp. 935–944. Lecture Notes on Computer Science, Springer-Verlag, Berlin, Heidelberg (2008)
11. Papa, J.P., Falcão, A.X., Suzuki, C.T.N.: Supervised pattern classification based on optimum-path forest. International Journal of Imaging Systems and Technology 19(2), 120–131 (2009)

12. Papa, J.P., Suzuki, C.T.N., Falcão, A.X.: LibOPF: A library for the design of optimum-path forest classifiers (2009), software version 2.0 available at `http://www.ic.unicamp.br/~afalcao/LibOPF`
13. Papa, J., Falcão, A.: A learning algorithm for the optimum-path forest classifier. In: Graph-Based Representations in Pattern Recognition. pp. 195–204. Springer Berlin/Heidelberg (2009)
14. Perricone, M.J., Dupont, J.N.: Effect of composition on the solidification behavior of several ni-cr-mo and fe-ni-cr-mo alloys. Metallurgical and Materials Transactions A. 37(4), 1267–1280 (2006)
15. Rocha, L.M., Cappabianco, F.A.M., Falcão, A.X.: Data clustering as an optimum-path forest problem with applications in image analysis. International Journal of Imaging Systems and Technology 19(2), 50–68 (2009)