This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

Improved Random Forest for Classification

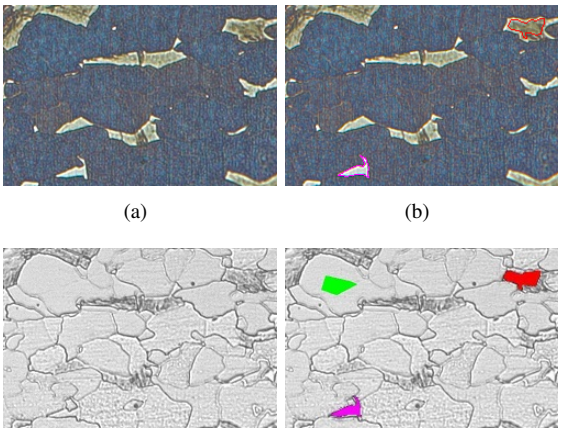
Angshuman Paul, Dipti Prasad Mukherjee, *Senior Member, IEEE,* Prasun Das, Abhinandan Gangopadhyay, Appa Rao Chintha and Saurabh Kundu

***Abstract*—We propose an improved random forest classifier that performs classification with minimum number of trees. The proposed method iteratively removes some unimportant features. Based on the number of important and unimportant features, we formulate a novel theoretical upper limit on the number of trees to be added to the forest to ensure improvement in classification accuracy. Our algorithm converges with a reduced but important set of features. We prove that further addition of trees or further reduction of features does not improve classification performance. The efficacy of the proposed approach is demonstrated through experiments on benchmark datasets. We further use the proposed classifier to detect mitotic nuclei in the histopathological datasets of breast tissues. We also apply our method on the industrial dataset of dual phase steel microstructures to classify different phases. Results of our method on different datasets show significant reduction in average classification error compared to a number of competing methods.**

***Index Terms*—Random forest, optimal number of trees, classification accuracy, feature reduction.**

I. Introduction

RANDOM forest has found its wide spread use in various applications [1]–[3]. The acceptability of random forest can be primarily attributed to its capability of efficiently handling non-linear classification task. Random forest is well-known for taking care of data imbalances in different classes [4], [5] especially for large datasets [6]. Due to its parallel architecture, random forest classifier is faster compared to other state-of-the-art classifiers. A large number of variants of random forest can be found in literature [7], [8]. According to [9], classification performance of random forest improves with increase in the number of trees [9]. But experimental evidences suggest that adding trees beyond certain pre-determined limit may not significantly improve the classification performance of random forest [10]. Some efforts have been made to find *apriori* the optimal number of trees in random forest [11], [12]. But these methods lack generalization with respect to dataset. Further, it has been shown that random forest may perform biased feature selection for individual trees [13]. As a result, an unimportant feature may be favored in a noisy feature set. Consequently, classification accuracy may degrade. So, an increased proportion of important features (i.e. removal of unimportant features) may have significant impact on the classification performance of random forest. A number of feature selection strategies can be found in the literature. In [14], the forest is grown using features that are likely to be important. But the decision of importance is taken based on the performances of different features in an initial forest. So, the problem of biased selection of features may take place.



(a) (b) (c) (d)

Figure 1: Sample steel micrographs using *le pera* etching ((a) original, (b) marked) and *nital* etching ((c) original, (d) marked) where (a) and (c) are identical samples. In (b) and (d): red-bainite, magenta-martensite. In (b): blue (background)-ferrite. In (d): green -ferrite.

We introduce an improved random forest (IRF) which takes care of feature selection and finds optimal number of trees simultaneously. IRF starts with a forest of small number of trees. The initial forest finds a small number of important features (dimensions of feature vector is referred to here as 'features'). Then at each pass, some more features are marked as important and some unimportant features are discarded using a novel condition. Based on the number of important and unimportant features, we calculate an upper bound in the number of trees to be added to the forest at a pass. We show that addition of trees satisfying this bound, ensures improvement in classification accuracy. This iterative process converges after a few passes. Meanwhile, most of the unimportant features are removed. After convergence, classification accuracy by the forest does not improve significantly by further addition of trees or further reduction of features. Thus the proposed forest provides optimal classification accuracy in terms of the number of trees and in terms of feature reduction. Notably, the number of trees in our method is not pre-determined like [10] for specific datasets. So IRF has low data dependence. IRF is fast and hence useful for industrial applications.

We apply the proposed classifier on a wide variety of datasets. First we perform experiments on several benchmark

A. Paul and D. P. Mukherjee are with Electronics & Communication Sciences Unit, Indian Statistical Institute, Kolkata, India. P. Das is with Statistical Quality Control & Operations Research Unit, Indian Statistical Institute, Kolkata, India. E-mail: {apaul_r, dipti, prasun}@isical.ac.in

A. Gangopadhyay is with School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ. E-mail: abhinandan.gangopadhyay@gmail.com

A. R. Chintha and S. Kundu are with Research and Development and Scientific Services, Tata Steel Limited, Jamshedpur, India. E-mail: {apparao, saurabhkundu}@tatasteel.com

Manuscript received June 5, 2017.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

datasets [15]–[21]. Then we take breast cancer datasets for performance evaluation [22]–[24]. There are several publicly available datasets where stained breast tissue is observed under microscope for possible cancer. Breast tissues contain mainly mitotic and non-mitotic nucleus. Here the goal is to find mitotic nucleus which may lead to uncontrolled cell division and subsequently lead to malignancy. From all possible segmented nuclei, we apply the proposed IRF to find the mitotic ones.

Finally we undertake an industrial application. We classify different phases of dual phase (DP) steel microstructures. The features for classification are extracted from the optical micrographs of the microstructures. Microstructure of DP steel is a mixture of ferrite and martensite phases. This combination provides high strength, low yield to tensile strength ratio, ductility and cold-formability [25], [26]. Some bainite may also form during the production of DP steel. The percentage of these three phases are manually calculated before industrial production. *Le Pera* etched optical micrographs of DP steel are used for manual calculation. In *le pera* etched images, ferrite appears blue, bainite appears dark and martensite appears white (see Fig. 1(a) and 1(b)). So, ferrite, bainite and martensite phases can be prominently identified from a *le pera* etched image of DP steel. However, *le pera* etching is extremely time consuming and requires human expertise. Hence, the manual method using *le pera* etching is unsuitable in an industry scenario.

Nital etching [27] is a faster alternative to *le pera* etching. But it is difficult to identify different phases from *nital* images (see Fig. 1(c)). Further the appearances of the phases in *nital* images may vary significantly across different samples due to a number of reasons. These include quality of the etching chemical, change in illumination and quality of the steel. So, manual calculation of phase percentage is not possible from *nital* etched samples. We use the proposed IRF for the classification of *nital* etched samples replacing time consuming manual inspection of *le pera* samples.

To the best of our knowledge, no other literature proposes a random forest where feature removal and tree additions are performed simultaneously ensuring improvement in classification accuracy. Furthermore, the proposed formulation of random forest is generic in nature. We show that the behavior of conventional random forest [9] can be explained as a special case of IRF. IRF works equally well for benchmark datasets, histopathological datasets and industrial datasets of steel microstructures. The rest of the paper is organized as follows: in Section II, we describe IRF in detail. The experimental details, results and related discussions are presented in Section III and finally we conclude the paper in Section IV.

II. Methods

First we briefly discuss the classical random forest [9]. Random forest [9] is composed of a pre-defined number of binary decision trees. Individual trees in the forest are grown using a bootstrap sample [9] from the training dataset. Consider that there are M features associated with each feature vector. During the growth of a random decision tree, a subset of f $(f < M)$ features are randomly chosen in each node. One out of these f features is selected for splitting the node.

The number of trees in a random forest is increased in an iterative fashion in the proposed improved random forest. Each of these iterations is referred as a construction pass. The process starts with an initial number of trees. Then at each construction pass, we update the list of important and unimportant features through following four steps. First we calculate the weights of different features and rank the features based on their weights (see Section II-A). Then we calculate a threshold weight. The features with weight below the threshold are subsequently removed. Next, from the set of remaining features, we mark some features as 'important' based on a novel criterion (see Section II-B). The rest of the features are marked as 'unimportant'. Note that, once a feature is marked to be important at a construction pass, it will remain important till the end and will not be removed in the subsequent passes. Once we have certain important and unimportant features, we formulate a theoretical bound of maximum number of trees to be added to the forest at that construction pass (see Section II-C). We show that if trees are added satisfying the bound, the classification accuracy of the forest certainly improves.

The construction passes are continued until a novel termination criterion is reached. Upon termination, the converged forest produces an optimal classification. The classification is optimal in the sense that adding more trees does not improve the classification performance significantly any further. Upon convergence, most (or all) of the unimportant features are removed. The classification performance is optimal also in the sense that further removal of features does not have any significant impact on the classification performance. Due to our formulation of the feature weight, important features tend to have higher weights. Note that we remove a feature only if the feature was not marked as 'important' in any of the previous passes and has a low weight (below the threshold) in the current pass. As a result, probability of discarding an important feature is reduced. The pipeline of the proposed method is presented in Fig. 2.

To enhance readability, a table of frequently used symbols along with their meanings is presented in Table I. In the next few subsections we discuss our algorithm in details with three major steps: feature ranking, finding important and unimportant features and finding the number of trees to be added. The feature ranking which is discussed next.

A. Feature Ranking

Consider an initial feature vector $\mathcal{F}_0(.)$. While growing a random tree, we use a entropy based measure for node splitting. Let us take a node i in a tree τ. Let the probability of class label c at this node be $p(c)$. Then entropy of that node is $E = \sum_{\forall c} p(c) \ln \frac{1}{p(c)}$. For splitting this node, we first choose a set (Λ) of f features randomly from $\mathcal{F}_0(.)$ without replacement. Assume that feature j is present in Λ and we split node i with feature j. Let the resultant left and right child of node i have entropy E_l and E_r respectively. Then, for node i, we define the quality of split by feature j as: $Q(i,j) = \exp\{-(E_l + E_r)\}$. The feature that provides highest quality of split, is chosen to split the node (we call it 'split feature').

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

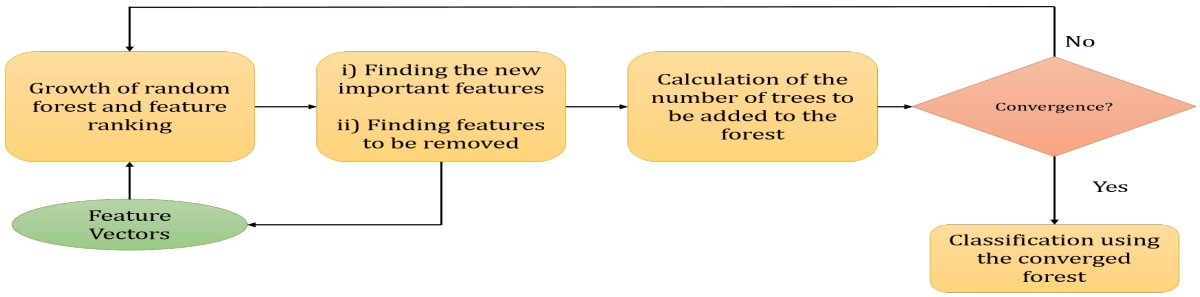


Figure 2: The pipeline of the proposed method.

Table I: List of the symbols and their meanings

Symbol	Definition
$\mathcal{F}_0(.)$	Initial feature vector
$\mathcal{F}_n(.)$	Feature vector at the n^{th} construction pass
Θ_n	Forest at the n^{th} construction pass
B_n	Number of trees at the n^{th} construction pass
u_n	Number of important features at the n^{th} construction pass
v_n	Number of unimportant features at the n^{th} construction pass
Γ_n	Bag of important features at the n^{th} construction pass
Γ'_n	Bag of unimportant features at the n^{th} construction pass
$w(j)$	Weight of feature j
$\mathcal{R}_n$	Features to be removed at the n^{th} construction pass
$\mathcal{A}_n$	Features to be newly marked as important at the n^{th} construction pass
Λ	Set of features for a node from which the split feature is selected
f	$\#\Lambda$, i.e. number of features from which one is selected for node splitting
q	Probability of selecting at least one important feature in Λ or minimum probability of good split in a node
r	Probability of selecting no important features in Λ
η_s	Strength of a forest
η_c	Value of correlation
N_{av}	Average number of nodes per tree
ρ	Probability that all the N_{av} pair of nodes in two trees have at least one feature in common in their corresponding Λs
η	Classification accuracy
η_μ	Overall recall

Let N be the total number of nodes in tree τ. We first assign a local weight to feature j w.r.t. tree τ:

$$w^\tau(j) = \frac{\sum_{i=1}^{N} Q(i,j)}{N}. \tag{1}$$

The higher the value of $w^\tau(j)$, the better is the quality of split by feature j in tree τ. We calculate weights of the trees based on out-of-bag error [28]. Let (δ^τ) be the out-of-bag error for tree τ. Then the normalized weight of tree τ is:

$$\gamma^\tau = \frac{1/\delta^\tau}{\max_\tau \ (1/\delta^\tau)}. \tag{2}$$

Higher value of γ^τ indicates lesser classification error by tree τ. Hence, features used for splitting the nodes of tree τ are more discriminative features. Using the local weight of feature and the weights of the trees, we calculate global weight of feature j:

$$w(j) = \frac{\sum_{\forall\tau} w^\tau(j)\gamma^\tau}{\max_j \sum_{\forall\tau} w^\tau(j)\gamma^\tau}. \tag{3}$$

A feature with higher value weight is more important for classification. Based on the global weights $w(j)$, we rank the features to find the important and unimportant features in the next section.

B. Finding Important and Unimportant Features

We do not know which and how many features are important (class discriminative). So we take a novel strategy to find the important features. Initially, from the ranked list, we mark top u_0 features as 'important' and rest of the features as 'unimportant'. Let Γ (initially Γ_0) be the bag of important features and Γ' (initially Γ'_0) be the bag of unimportant features. These bags of features are updated at every construction pass. Consider the n^{th} pass. Let μ_n and σ_n be the mean and standard deviation of the global weights of the features present in the bag of unimportant features Γ'_n. Then we put the features with global weight $< (\mu_n - 2\sigma_n)$ in a set $\mathcal{R}_n$. We discard the features in $\mathcal{R}_n$. Now let there be some feature j with weight $w(j)$ in the bag of unimportant feature Γ'_n. Assume that $w(j)$ is greater or equal to minimum of feature weights in Γ_n. Then we mark feature j as important. We promote j from Γ'_n to Γ_n. So, a feature j in Γ'_n is newly marked as important if

$$w(j) \geq \min_k \ w(k); \ j \in \Gamma'_n, \ k \in \Gamma_n. \tag{4}$$

The modified feature bags are denoted as Γ_{n+1} and Γ'_{n+1}. Let $\mathcal{A}_n$ be the set of newly found important features. Then:

$$\Gamma_{n+1} = \Gamma_n + \mathcal{A}_n \ , \ \Gamma'_{n+1} = \Gamma'_n - \mathcal{R}_n - \mathcal{A}_n. \tag{5}$$

Note that once a feature is marked to be important, it is never removed in subsequent passes. Thus the probability of discarding an important feature is reduced. Importance of a feature may not be evident immediately at a split. So we do not discard any feature at individual nodes during the growth of a tree. Thus the probability of discarding an important feature is further reduced. Let u and v denote the number of important and unimportant features respectively in a construction pass according to our algorithm. Then

$$\Delta u = \#\Gamma_{n+1} - \#\Gamma_n \ , \ \Delta v = \#\Gamma'_{n+1} - \#\Gamma'_n. \tag{6}$$

Using u, v, Δu and Δv, we calculate the number of trees to be added to the forest as detailed next.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

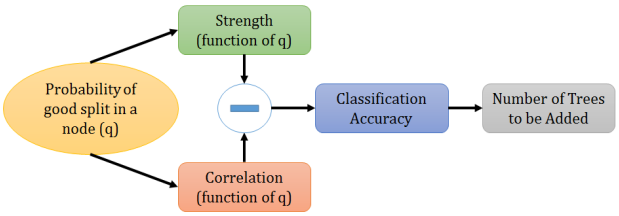


Figure 3: The steps of finding the number of trees to be added to the forest.

C. Finding the Number of Trees to be Added

To find the number of trees to be added, we first define two quantities that controls the classification performance of random forest. According to [9], these two quantities are strength and correlation. Classification accuracy is defined based on strength and correlation. We find the number of trees to be added using the formulation of classification accuracy. The steps for finding the number of trees are presented in Fig. 3. In the next few paragraphs, we perform a detailed analysis. This analysis requires the calculation of probability of good split. Based on the probability of good split, we formulate strength and correlation.

1) Probability of Good Split: Probability of good split is the probability that a node is split by an important feature. According to [29], a good split creates child nodes with more homogeneity compared to the parent node. A good split in node i is possible only if at least one important feature (i.e. a feature from Γ) is present in corresponding Λ. Recall that features in Λ are selected randomly from the set of total features i.e. $\Gamma \cup \Gamma'$. There is possibility that some feature, present in the bag of unimportant features Γ' might turn out to be important in the subsequent construction passes. Hence, if we select features from only the bag of important features Γ, it will lead to greedy selection and we will miss some potential important features. Hence, we choose the features in Λ from both the bags of important and unimportant features.

The minimum probability of a good split in node i of tree τ is same as the probability of picking up at least one important feature in corresponding Λ. Let us denote this probability as q. Hence q is the minimum probability of good split in a node. Then $r = (1-q)$ denotes the probability that no important feature is present in Λ. Bag $\Gamma'(\#\Gamma' = v)$ contains all the unimportant features and may also contain some important features (which may be promoted to bag Γ in subsequent passes). So, the maximum value of r is:

$$r = 1 - q = \frac{\binom{v}{f}}{\binom{u+v}{f}}. \tag{7}$$

It is obvious that if $v < f$, at least one important feature will always be selected in Λ. So if $v < f$, $q = 1$ and $r = 0$.

Consider a forest of B trees. Let us denote $q_u = \frac{\partial q}{\partial u}$ and $q_v = \frac{\partial q}{\partial v}$. Then from (7), we have $q_u = -\frac{\partial r}{\partial u}$ and $q_v = -\frac{\partial r}{\partial v}$. We approximate $\frac{\partial r}{\partial u}$ by taking discrete difference of r w.r.t. u keeping B, v constant. Similarly, we also approximate $\frac{\partial r}{\partial v}$. So $q_u \approx -\left(\frac{\Delta r}{\Delta u}\right)_{B,v}$ and $q_v \approx -\left(\frac{\Delta r}{\Delta v}\right)_{B,u}$. Then using (7):

$$\begin{aligned} q_u \approx -\left(\frac{\Delta r}{\Delta u}\right)_{B,v} &= \frac{v!\,(u+v-1-f)!\,f}{(v-f)!\,(u+v)!}, \\ q_v \approx -\left(\frac{\Delta r}{\Delta v}\right)_{B,u} &= -\frac{(v-1)!\,(u+v-1-f)!\,uf}{(v-f)!\,(u+v-1)!\,(u+v)}. \end{aligned} \tag{8}$$

So, $q_u > 0$ and $q_v < 0$. Next we define strength and correlation using q. Definitions of strength and correlation are based on the fact that average number of nodes per tree (N_{av}) does not change significantly as the number of trees in the forest is increased (See Appendix A for the proof). Using this fact, we define the strength and correlation next.

2) Strength: Consider a tree with N_{av} number of nodes. We calculate the probability that good split occurs at each node of this tree. The minimum probability of good split in a node is q. Since the tree has N_{av} number of nodes, probability of good splits in all the nodes is $q^{N_{\text{av}}}$. We assume minimum classification accuracy by the tree to be proportional to $q^{N_{\text{av}}}$. Consider a forest of B trees. According to [9], the strength of a forest is dependent on the minimum classification accuracy of individual trees. Hence, we define the strength of the forest (η_s) as the probability that all the nodes in at least one tree has good splits:

$$\begin{aligned} \eta_s &= \sum_{\tau=1}^{B} \binom{B}{\tau} \left(q^{N_{\text{av}}}\right)^{\tau} \left(1 - q^{N_{\text{av}}}\right)^{B-\tau} \\ &= 1 - \left(1 - q^{N_{\text{av}}}\right)^{B}. \end{aligned} \tag{9}$$

3) Correlation: Next we calculate correlation between any two trees in the forest. Correlation is a measure of similarity between the trees. For random forest, correlation between trees is dependent on the features used at different nodes of those trees [9]. At any point of time, at most $B/2$ tree pairs may exist in a forest with B trees. Consider a pair of trees τ_1 and τ_2. For splitting node i in τ_1 we first randomly choose f features in $\Lambda_i^{\tau_1}$. This selection can be made in $\binom{u+v}{f}$ ways. Same is the case for $\Lambda_i^{\tau_2}$ (set of features for node i in tree τ_2). So, the probability (ρ') that at least one feature is common in $\Lambda_i^{\tau_1}$ and $\Lambda_i^{\tau_2}$ is:

$$\rho' = 1 - \frac{\binom{u+v}{f}\binom{u+v-f}{f}}{\binom{u+v}{f}\binom{u+v}{f}} = 1 - \frac{\binom{u+v-f}{f}}{\binom{u+v}{f}}. \tag{10}$$

Then the probability (ρ) that all the N_{av} pairs of nodes in tree τ_1 and τ_2 have at least one feature in common, is given by:

$$\rho = (\rho')^{N_{\text{av}}} = \left(1 - \frac{\binom{u+v-f}{f}}{\binom{u+v}{f}}\right)^{N_{\text{av}}}. \tag{11}$$

Notice that $\rho \ll 1, \forall\,(u,v)$. Hence $\frac{\partial \rho}{\partial u} \to 0$ and $\frac{\partial \rho}{\partial v} \to 0$. Consequently, we define correlation (η_c) as the probability that at least one pair of nodes (from two different trees) have at least one feature in common:

$$\eta_c = \sum_{\tau=1}^{B/2} \binom{B/2}{\tau} \rho^{\tau} (1-\rho)^{(B/2)-\tau} = 1 - (1-\rho)^{B/2}. \tag{12}$$

Based on strength and correlation, next we define classification accuracy.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

4) Classification Accuracy and Number of Trees: Classification accuracy (η) of the proposed forest is defined using strength and correlation as:

$$\eta = \lambda\left(\eta_s - \eta_c\right) = \lambda\left(\left(1-\rho\right)^{B/2} - \left(1 - q^{N_{\text{av}}}\right)^{B}\right), \quad (13)$$

where λ is a constant. Notice that q and ρ are functions of u and v. N_{av} is a constant. So, η is a function of u, v and B. Hence we can write:

$$d\eta = \lambda\left[\frac{\partial\left(\eta_s - \eta_c\right)}{\partial B}dB + \frac{\partial\left(\eta_s - \eta_c\right)}{\partial u}du + \frac{\partial\left(\eta_s - \eta_c\right)}{\partial v}dv\right]. \quad (14)$$

Let us assume $\nu = \frac{\partial(\eta_s - \eta_c)}{\partial B}$. Then from (13), we obtain:

$$\begin{aligned}\nu = \frac{\partial\left(\eta_s - \eta_c\right)}{\partial B} &= \frac{\left(1-\rho\right)^{B/2}}{2}\ln\left(1-\rho\right) \\ &\quad - \left(1 - q^{N_{\text{av}}}\right)^{B}\ln\left(1 - q^{N_{\text{av}}}\right). \quad (15)\end{aligned}$$

From (13), we find:

$$\begin{aligned}\frac{\partial\left(\eta_s - \eta_c\right)}{\partial u} &= -\frac{B}{2}\left(1-\rho\right)^{(B/2)-1}\frac{\partial\rho}{\partial u} \\ &\quad + BN_{\text{av}}q^{N_{\text{av}}-1}\left(1 - q^{N_{\text{av}}}\right)^{B-1}\frac{\partial q}{\partial u}, \\ \frac{\partial\left(\eta_s - \eta_c\right)}{\partial v} &= -\frac{B}{2}\left(1-\rho\right)^{(B/2)-1}\frac{\partial\rho}{\partial v} \\ &\quad + BN_{\text{av}}q^{N_{\text{av}}-1}\left(1 - q^{N_{\text{av}}}\right)^{B-1}\frac{\partial q}{\partial v}. \end{aligned} \quad (16)$$

Let $l = BN_{\text{av}}q^{N_{\text{av}}-1}\left(1 - q^{N_{\text{av}}}\right)^{B-1}$. When $q < 1$, we have $l > 0$. Recall that $\frac{\partial\rho}{\partial u} \to 0$ and $\frac{\partial\rho}{\partial v} \to 0$. Hence we represent (16) as:

$$\frac{\partial\left(\eta_s - \eta_c\right)}{\partial u} = l\frac{\partial q}{\partial u}, \ \frac{\partial\left(\eta_s - \eta_c\right)}{\partial v} = l\frac{\partial q}{\partial v}. \quad (17)$$

Since $q_u = \frac{\partial q}{\partial u}$, $q_v = \frac{\partial q}{\partial v}$, using (15) and (17), we rewrite (14):

$$d\eta \approx \lambda\left(\nu\Delta B + lq_u\Delta u + lq_v\Delta v\right). \quad (18)$$

We need $d\eta > 0$ to improve the classification accuracy. Recall that $q_u > 0$, $q_v < 0$ and $l > 0$. So, to improve accuracy, we make $\Delta v < 0$. This supports the fact that reduction of unimportant feature would help in better classification. Since $q_u > 0$, we need to make $\Delta u \geq 0$ to improve η. Finally, from (15) notice that the value of ν may be positive or negative depending on the values of q, ρ and B. Then, to ensure $d\eta > 0$:

$$\begin{aligned}&\left|\nu\ \Delta B\right| < \left|lq_u\Delta u + lq_v\Delta v\right| \\ \implies &\left|\Delta B\right| < \left|\frac{l\left(q_u\Delta u + q_v\Delta v\right)}{\nu}\right|. \end{aligned} \quad (19)$$

In order to find ΔB, we calculate q_u and q_v numerically from (8). It has been shown that adding more trees almost surely improves the classification performance [9]. So, we never make ΔB negative. Based on the above arguments, we design the improved random forest (IRF) algorithm.

D. The IRF Algorithm

We first grow an initial forest Θ_0 with B_0 number of trees and initial feature vector $\mathcal{F}_0(.)$. After the feature ranking using (3), we mark top u_0 number of features to be important (we take $u_0 = \sqrt{\#\mathcal{F}_0(.)}$). We keep the important features in bag Γ_0 and rest of the features in bag Γ_0'. Let the mean and standard deviation of feature weights in Γ_0' be μ_0 and σ_0 respectively. Then $\mathcal{R}_0$ is the initial subset of features from Γ_0' whose weights are less than $(\mu_0 - 2\sigma_0)$ (In case there is no such feature, $\mathcal{R}_0$ contains the feature with smallest global weight in Γ_0'). We remove features of $\mathcal{R}_0$ from Γ_0'. The feature vector with reduced set of feature is $\mathcal{F}_1(.) = \mathcal{F}_0(.) - \mathcal{R}_0$. Next we calculate Δu and Δv using (6). Based on these values, we find the number of trees to be added (ΔB) to the forest using (19). Then a forest Θ_1 is grown afresh with $B_0 + \Delta B$ and feature vectors $\mathcal{F}_1(.)$.

At any pass n (after the growth of forest Θ_n with B_n trees and reduced feature set $\mathcal{F}_n(.)$), we first rank the features. Then we find the set of new important features $\mathcal{A}_n$ and the set of features to be reduced $\mathcal{R}_n$. We obtain the feature vector with reduced features $\mathcal{F}_{n+1}(.) = \mathcal{F}_n(.) - \mathcal{R}_n$ and update the feature bags using (5). Next we calculate Δu and Δv from (6) and ΔB using (19). Finally we grow a forest Θ_{n+1} with $B_n + \Delta B$ number of trees and feature vectors $\mathcal{F}_{n+1}(.)$. Algorithm 1 shows the proposed method. Our formulation of random forest is generic. In Appendix B, we show that the performance of random forest proposed by Breiman [9] can be explained as a special case of our model. Next we show that the proposed random forest surely converges.

E. Convergence

As we have seen, v is reduced to improve η. When $v < f$, from (7), we have $q = 1$ (constant). So, $q_u = \frac{\partial q}{\partial u} = 0$ and $q_v = \frac{\partial q}{\partial v} = 0$. Then, from (18) and (19), we find that $d\eta \approx \lambda\left(\frac{(1-\rho)^{B/2}}{2}\ln\left(1-\rho\right)\right)\Delta B$. From (9), notice that, at this point $\eta_s = 1$. So, when $v < f$, strength saturates. Now since $\rho < 1$, further increase in B would make $d\eta < 0$. This implies, further increase in number of trees only increases the correlation. As a result classification accuracy decreases. So, we do not increase the number of trees any further ($\Delta B = 0$). Then we obtain $d\eta \approx 0$. So, classification accuracy becomes constant at this point. Reduction of feature or addition of trees does not improve η any further. So, we terminate the algorithm once we have $v < f$. We use this converged forest for testing.

From line 10 of Algorithm 1, notice that at each pass of the algorithm, some features from Γ' are definitely removed. Some features from Γ' may also move to the set of important features Γ (as evident from line 14 of Algorithm 1) following (4). So, at each pass, v is definitely reduced. Consequently, it is guaranteed that at some pass n, we will have $v < f$. This ensures convergence of the proposed algorithm (from line 8 of Algorithm 1).

F. Testing

The converged random forest from Algorithm 1 is used for classifying unlabeled test data. For classification, the test data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

Algorithm 1 Improved Random Forest

procedure INITIALIZE(Θ_0)
  Grow initial forest Θ_0 with B_0 random trees and feature vector $\mathcal{F}_0(.)$.
  Calculate weight $w(.)$ and rank all the features in $\mathcal{F}_0(.)$ using (3).
  From the ranked list of features, put top $u_0 = \sqrt{\#\mathcal{F}_0(.)}$ features in Γ_0.
  Put rest $v_0 = \#\mathcal{F}_0(.) - u_0$ features in Γ'_0.
  n is the number of pass. Initialize $n = 0$.
end procedure
while $v_n \geq f$ **do**
  Compute mean (μ_n) and standard deviation (σ_n) of feature weights in Γ'_n.
  Find $\mathcal{R}_n = \{j\}\,; \forall j \in \Gamma'_n : w(j) < (\mu_n - 2\sigma_n)$; if no such j exists, $\mathcal{R}_n = \{j\}\,, j \in \Gamma'_n : w(j) = \min_{k \in \Gamma'_n} w(k)$.
  From the bag of unimportant features, find the set of features $(\mathcal{A}_n)$ whose weights are larger than the minimum of the weights of important features. So, $\mathcal{A}_n = \{j\} : w(j) \geq \min_k w(k);\ \forall j \in \Gamma'_{n+1},\ k \in \Gamma_{n+1}$.
  Find $\mathcal{F}_{n+1}(.) = \mathcal{F}_n(.) - \mathcal{R}_n$.
  Find $\Gamma_{n+1} = \Gamma_n + \mathcal{A}_n$ and $\Gamma'_{n+1} = \Gamma'_n - \mathcal{R}_n - \mathcal{A}_n$.
  $u_{n+1} = \#\Gamma_{n+1}$ and $v_{n+1} = \#\Gamma'_{n+1}$. Calculate $\Delta u = u_{n+1} - u_n$ and $\Delta v = v_{n+1} - v_n$.
  Find ΔB using (19); $B_{n+1} = B_n + \Delta B$.
  Grow forest Θ_{n+1} using B_{n+1} trees and feature vector $\mathcal{F}_{n+1}(.)$.
  Calculate weight $w(.)$ and rank all the features in $\mathcal{F}_{n+1}(.)$ using (3).
  $n = n + 1$.
end while

is entered at the root node of each tree of the converged forest. The weights of the trees are determined by (2). As the data reaches the leaf node of the trees, a weighted voting from the trees is used for determining the class label of the test data. Let the converged forest contain B^* number of trees and $\gamma^*(\tau)$ denote the weight of tree τ in the converged forest. If $p^*_\tau(c)$ denotes the probability of class label, predicted by tree τ for the input test data, then the actual class label (c^*) for the input test data is given by:

$$c^* = \underset{c}{\operatorname{argmax}} \sum_{\tau=1}^{B^*} \gamma^*(\tau) p^*_\tau(c), \tag{20}$$

Next we present the results using the proposed method.

III. Experimental Results

We design improved random forest (IRF) for classification tasks. We have few parameters to tune for our IRF classifier. We first demonstrate the effectiveness of the proposed method on a number of benchmark datasets. Note that we do not perform any pre-processing on the datasets. Then we perform classification of histopathological images from breast tissues. Finally the efficacy of the proposed method is established in steel microstructure datasets.

Table II: Description of the datasets and the number of trees used (#Feat and #Cls indicate the number of features and the number of classes respectively)

Dataset	#Train	#Test	#Feat	#Cls	#Trees used
MNIST	50000	10000	784	10	80
MNIST-Rot	12000	50000	784	10	60
MNIST-Bkg	12000	50000	784	10	65
MNIST-RandB	12000	50000	784	10	70
MNIST-RotB	12000	50000	784	10	60
USPS	7291	2007	256	10	60
ISOLET	6238	1559	618	26	85
G50c	50	500	50	2	45
Letter	16000	4000	16	26	75
Madelon	2000	600	500	2	65
Spam Base	3681	920	57	2	95
Gamma	15216	3804	11	2	80
Page Blocks	4378	1095	10	5	90

There are two main reasons of using breast cancer and steel microstructure datasets. First, for the breast cancer datasets, the size of the training dataset is small. As a consequence, it is difficult to get a reliable result for these datasets using stand-alone classifier. On the contrary, random forest is an ensemble of classifiers. Each tree acts as a weak classifier. Each tree is grown using separate bootstrap samples drawn from the training data. Hence every tree finds different decision boundaries and produces different classification results on test data. The final decision on classification is taken based on majority voting or averaging of outputs of ensemble of weak classifiers. The averaging helps reduce classification error which might have occurred due to a stand-alone classifier.

Second, due to various factors in data acquisition, data of same class may appear differently in a dataset. As a result, intra-class variations for both breast cancer and steel microstructure datasets are significant. Variation in H&E staining in case of mitotic cell images or variation in expert-dependent etching in case of steel micro-graph images are the causes of intra-class variations. Again ensemble classifiers like random forest are known to tackle such variations. Our choice of these datasets and the performance of our proposal on these datasets should show the efficacy of use of ensemble classifiers. Next we present the results of the proposed technique in different datasets.

A. Benchmark Datasets

1) Datasets & Parameters: We use several benchmark datasets for our experiments. Those are:

- MNIST [15]: This dataset consists of images of handwritten digits (0-9). From the images, 784 features are extracted for each sample in the database. The dataset contains 60000 training and 10000 test samples.
- Variants of MNIST [16]: We also use a number of variants of MNIST. In particular we use rotated MNIST (MNIST-Rot), MNIST with background image (MNIST-Bkg), MNIST with random background (MNIST-RandB) and rotated MNIST with background images (MNIST-RotB). All these datasets contain 12000 training images and 50000 test images.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

Table III: Average error percentage (mean±sd) for different methods in benchmark datasets

Dataset	ADA	RUS	RF	SVM	IRF
MNIST	11.3±0.9	12.1±1.3	11.2±1.7	10.7±1	7.6±1.1
MNIST-Rot	21.4±1.5	20.7±1.8	19.8±1.2	21.2±1.5	17.7±0.9
MNIST-Bkg	27.2±1.8	25.7±1.3	24.1±1.3	23.8±1.6	22.1±1.4
MNIST-RandB	18.9±1.8	19.1±2.9	19.4±2.3	17.3±2.1	17.6±1.7
MNIST-RotB	61.1±4.3	59.9±5.2	59.6±3.7	57.7±4.6	54.6±3.3
USPS	5.9±0.3	6.1±0.2	6.1±0.3	5.9±0.6	5.5±0.3
ISOLET	13.2±1.1	12.1±1.3	11.7±0.9	7.3±1.3	6.1±1.1
G50c	19.1±1.5	18.9±1.2	19.1±1.3	19.2±1.1	18.7±1.2
Letter	4.9±0.2	4.8±0.3	4.8±0.2	4.1±0.5	3.7±0.1
Madelon	23.2±4.1	20.2±1.3	15.7±2.8	7.2±2.5	6.3±1.3
Spam Base	5.1 ± 0. 6	5.2 ± 0.5	4.8 ± 0.2	4.9 ± 0. 4	3.9 ± 0.2
Gamma Telescope	13.3 ± 1.2	12.7 ± 0.7	12.1 ± 1.1	13.5 ± 0.8	11.51 ± 0.6
Page Blocks	3.1 ± 0.3	3.3 ± 0.1	2.9 ±0.3	2.5 ± 0.4	2.1 ± 0.2

- ISOLET [17]: It is a database of the spoken letters of the alphabet. Audio data samples are encoded into 617 dimension feature vectors. The training and test datasets contain 6238 and 1559 samples respectively.
- USPS [18]: It is a dataset of images of handwritten digits (0-9). The numbers of training and test data are 7291 and 2007 respectively. Each data point is a 256 dimensional feature vector.
- G50c [19]: It is an artificial data set generated from a standard normal multi-variate Gaussian. In G50c, the labels correspond to the Gaussians, and the means are located in 50-dimensional space such that the Bayes error is 5%.
- Letter [20]: It is a database of 26 capital letters of English alphabet. The character images are based on 20 different fonts and each letter within these 20 fonts are randomly distorted to produce a file of 20,000 unique data points.
- Madelon [21]: This is an artificial dataset with 2000 training samples and 600 test samples. Each sample is a 500 dimensional feature vector.
- Spam Base [30]: This dataset have features extracted from emails in order to classify the emails as spam or non-spam.
- Gamma Telescope [30]: Each data point in this dataset contains features extracted from high energy particles. The goal is to classify the particles as gamma or hadron.
- Page Blocks [30]: Each data point in this dataset concerns one block in a document. The goal is to classify the blocks in different categories like text, graphics etc.

The details of the dataset is presented in Table II. For classification, we need to select only one parameter f (cardinality of the set of possible split features for each node). According to [9], we take f to be the square root of initial number of features as presented in Table II.

2) Performance Measures: We first evaluate the performance of the proposed method in ten benchmark classification datasets. For each of these datasets, we first calculate the average error using the converged forest of the proposed method. The number of trees in the converged forests have been presented in Table II. For our experiments, we take a dataset and run our method 10 times. We report the mean±sd of average error percentage on each dataset. The average error using the proposed method can be found in Table III. Using a computer with Intel Xeon processor and 16 GB memory, the maximum training time for benchmark datasets is $\sim 200s$ and maximum testing time is $\sim 40s$.

3) Comparisons: Next we compare the performances of the proposed method with other state-of-the-art classification approaches. We mainly use tree based methods as our competitors. So, for each dataset, we first find how many trees are required in our IRF method. Then using that many trees, we run the competing methods. As an example, notice from Table II that MNIST dataset requires 80 trees for the proposed IRF method. Hence all the competing tree based methods are run with 80 trees for MNIST dataset. In particular we compare the performance of the proposed IRF method with AdaBoost [31] classifier. Since we are mostly dealing with classification problems with more than two classes, we use AdaBoost.M2 variant (ADA). Data imbalances in different classes is a major challenge in our problem. The RUSBoost (RUS) algorithm [32] is known to handle data imbalances efficiently. So, we also compare our method with RUSBoost classifier. Next, we show the comparison of the proposed method with conventional random forest (RF) [9]. We also compare our method with support vector machine (SVM) with radial basis function (RBF) kernel. However for SVM we have set two parameters: gamma (inverse of standard deviation radial basis function) and cost. For our experiments, we take the value of gamma as 0.8 and value of cost as 15000. These values are set through cross validations.

The comparative performances can be found in Table III. Notice that for most of the datasets, our method out-performs its competitors. The performance of the proposed method is not the best among the competing methods for MNIST-Rot and MNIST-RandB datasets. But still our method is better than most other methods in these datasets too. Notice the margin of improvement of our method from its closest competitor. This margin is as high as 12.5% for Madelon and 4.8% for MNIST-Rot. Thus we find that for most of the benchmark datasets, our method outperforms all of its competitors.

B. Breast Cancer Datasets

1) Datasets & Parameters: We take the mitosis detection problem for breast cancer datasets of histopathological images. There are three publicly available such datasets, namely: MITOS [22], AMIDA [23] and MITOS-ATYPIA [24]. These

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

Table IV: Average recall ($\overline{\text{Re}_c}$), precision ($\overline{\text{Pr}_c}$), F-measures (F_c) and overall recall (η_μ) for different breast cancer datasets after 5-fold cross validations

Dataset	MITOS		AMIDA		MITOS-ATYPIA	
	Mitotic	Non-Mitotic	Mitotic	Non-Mitotic	Mitotic	Non-Mitotic
$\overline{\text{Re}_c}$	0.83	0.85	0.78	0.72	0.76	0.74
$\overline{\text{Pr}_c}$	0.85	0.85	0.79	0.83	0.84	0.77
F_c	0.84	0.85	0.78	0.77	0.80	0.75
η_μ	0.84		0.75		0.75	

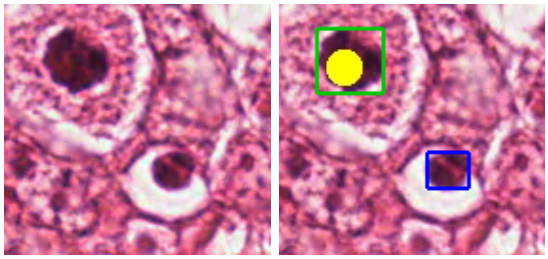
datasets contain hematoxylin and eosin (H&E) stained images of breast tissues. Two types of nucleus are present in the tissues: mitotic and non-mitotic. The goal of the problem is to identify the mitotic nuclei from these images. The stained breast tissue is scanned at $40\times$ magnification and a spatial resolution of almost 0.25 μm/pixel. Out of these three datasets, MITOS contain only 35 training images with 233 mitotic cells. AMIDA contains 311 training images with 550 mitotic cells. MITOS-ATYPIA on the other hand is a larger dataset with 1200 images and 749 mitotic cells.

For our experiments, we first apply segmentation technique proposed in [8] in the stained images of breast tissue to extract all the candidate nuclei (mitotic and non-mitotic) present in the images. Then we obtain 604 features for each nucleus by applying the feature extraction technique of [8]. Using these features, we have to classify the segmented nuclei in two classes: mitotic and non-mitotic. For these datasets, we take $f = \sqrt{604} \approx 25$.

2) Performance Measures: The classification problem for breast cancer datasets (MITOS, AMIDA and MITOS-ATYPIA) involves classifying the nuclei into two classes, namely: mitotic and non-mitotic. Let D_c be the number of ground truth data points corresponding to class c. Then for class c, we find the number of true positives (TP_c), number of false positives (FP_c) and number of false negatives (FN_c) [24]. For class c, we calculate the recall (Re_c), precision (Pr_c) and F-measure (F_c). We also define overall recall (η_μ) as the average value of recall over all the classes.

We perform 5-fold cross validations in each of the datasets to evaluate the results. For each class c, we find average values of recall ($\overline{\text{Re}_c}$) and precision ($\overline{\text{Pr}_c}$) across the 5 folds of cross validation. Subsequently we calculate the F-measure (F_c) for class c using $\overline{\text{Re}_c}$ and $\overline{\text{Pr}_c}$. The average recall, precision and F-measures for different classes in different datasets have been reported in Table IV. Notice that we achieve F-measure value of at least 0.75 for mitotic class in all the datasets. This result is particularly significant since the goal in breast cancer datasets is to identify the mitotic nuclei accurately. We also obtain overall recall (η_μ) of at least 0.75. Results on a sample region of histopathological image has been presented in Fig. 4. Using a computer with Intel Xeon processor and 16 GB memory, the training times for MITOS, AMIDA and MITOS-ATYPIA are $\sim 600s$, $\sim 800s$ and $\sim 1500s$ respectively. The testing times are $\sim 40s$, $\sim 50s$ and $\sim 90s$ respectively.

3) Comparisons: Similar to the benchmark datasets, we compare our method with AdaBoost.M2 (ADA), RUSBoost (RUS), random forest (RF) [9] and support vector machine (SVM) with RBF kernel [33]. We find that for MITOS, AMIDA and MITOS-ATYPIA, our method needs 65, 80 and 85 trees respectively. Accordingly we take 65, 80 and 85 trees for the competing methods for MITOS, AMIDA and MITOS-ATYPIA datasets respectively. However, for SVM, gamma is taken to be 0.75 with cost 9000. We compare the performances using F-measure for mitotic nuclei (F_c) for all the methods. The results are presented in Table V. Notice that our method out-performs most of its competitors for all the datasets in terms of F-measure for mitotic nuclei. We also compare the overall recall for the three breast cancer datasets using different methods. The results can be found in Table VI. Notice that the proposed IRF out-performs all of its competitors in most of the datasets. Our method also beats deep neural network based approach proposed in [34] (with F measure of 0.782 for MITOS dataset) and convolutional neural network based approach of [35] (with F measure of 0.659 for MITOS dataset) in terms of F measure. Thus our algorithm is found to be superior compared to a number of competing approaches for breast cancer datasets.



(a) (b)

Figure 4: (a) Sample region on histopathological image of breast tissue and (b) Detected mitotic (green box), non-mitotic (blue box) and ground truth mitotic (yellow region) cells.

Table V: F-measure for detection of mitotic nuclei for different methods in breast cancer datasets

Dataset	ADA	RUS	RF	SVM	IRF
MITOS	0.77	0.73	0.79	0.83	0.84
AMIDA	0.75	0.76	0.77	0.77	0.78
MITOS-ATYPIA	0.71	0.76	0.8	0.81	0.8

Table VI: Overall recall for different methods in breast cancer datasets

Dataset	ADA	RUS	RF	SVM	IRF
MITOS	0.72	0.75	0.77	0.81	0.84
AMIDA	0.71	0.73	0.72	0.73	0.75
MITOS-ATYPIA	0.72	0.74	0.75	0.78	0.75

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

Table VII: Average recall ($\overline{\text{Re}_c}$), precision ($\overline{\text{Pr}_c}$) and F measures (F_c) for different classes in steel microstructure after 5-fold cross validations

Class	$\overline{\text{Re}_c}$	$\overline{\text{Pr}_c}$	F_c
Bainite	0.97	0.94	0.95
Martensite	0.89	0.96	0.92
Ferrite	0.98	0.97	0.97

C. Steel Microstructure Datasets

1) Datasets & Parameters: We focus on classifying different phases of dual phase (DP) steel microstructures. The images for our experiments have been captured using Leica DM6000 M microscope at $500\times$ magnification. The data have been prepared at Tata Steel, India. For ground truth preparation, one metallurgical expert marked some candidate regions from each optical micrograph of *nital* etched DP steel. Then two metallurgical experts classified those marked regions into one of the three (bainite, martensite, ferrite) phase classes. For this manual decision making, the experts took help of the corresponding *le pera* etched images. If the two experts agree at the class label of a particular region, we keep the pixels of that region for our experiments. If the experts have a conflict of opinion over the class of a marked region, we do not use the pixels from that region for our experiments. In total, we use more than half a million (506670) pixel samples from bainite, martensite and ferrite classes for our experiments. Out of these samples, 245155 samples are bainite, 114785 samples are martensite and 146730 samples are ferrite.

For each pixel, we evaluate three different types of features. First we take a $W \times W$ window around a pixel (x, y). Let us call this region $R(x, y)$. We divide the entire range of gray levels in b equal divisions. Then we calculate the number of contour lines per unit area (pixel square) in $R(x, y)$ at each of the b levels. So, for pixel (x, y), we get a b dimensional histogram (H_1) of contour lines per unit area. Next, for each pixel (x, y), we sub-divide $R(x, y)$ in overlapping sub-regions of size $W' \times W'$. We evaluate entropy of each such sub-region. Then we plot a histogram (H_2) of all the entropy values of $R(x, y)$ using b' number of bins. Finally, for each $W' \times W'$ sub-region, we evaluate mean and variance of pixel intensities. Using these values, we plot a 2-D mean-variance histogram (H_3) of $R(x, y)$ with $b \times b$ number of bins. We construct the feature vector $\mathcal{F}$ for pixel (x, y) concatenating the three histograms H_1, H_2 and H_3. So, our feature vector contains a total of $b + b' + (b \times b) = b' + b(1 + b)$ features.

For selecting W, we consider the fact that ferrite regions are the largest among the three phases in DP steel samples. However, bainite and martensite regions occur as islets (see Fig. 1). From domain knowledge, we find that typical ferrite and martensite regions do not have an area < 950 pixels. So, we take $W = 31$. For W', we choose the smallest possible option i.e. $W' = 3$. Because, choosing a small W' enables us to monitor even the finest variation of entropy, mean and variance in a phase region. The bin sizes b and b' are decided through cross validations (For details, see Appendix C). In particular, we choose $b = 14$ and $b' = 110$. Initially we have a 320 dimensional feature vector $\mathcal{F}_0(.)$ for each pixel.

Table VIII: F measures (F_c) of different classes in steel microstructure and overall F measures after 5-fold cross validations in different subsets

Class	SS1	SS2	SS3	SS4	SS5	mean±sd
Bainite	0.94	0.95	0.97	0.95	0.97	0.95±0.01
Martensite	0.91	0.9	0.92	0.88	0.91	0.90±0.01
Ferrite	0.96	0.97	0.98	0.98	0.97	0.97±0.008
Overall	0.93	0.94	0.95	0.93	0.95	0.94±0.01

Table IX: Overall recall (η_μ) for different subsets

Subsets	SS1	SS2	SS3	SS4	SS5	mean±sd
η_μ	0.95	0.95	0.94	0.93	0.94	0.944±0.0089

For classification, however we need to select only one parameter $f = \lceil \sqrt{\#\mathcal{F}_0(.)} \rceil = 18$. Typically, random forests require number of trees as input parameter. We propose a formulation for the initial number of trees (B_0) (see Appendix D). Based on this, we find that $B_0 > 19$ for our problem. So, we take $B_0 = 20$ for our experiments.

2) Performance Measures: We evaluate the performance of the proposed method. Let TP_c, FP_c and FN_c be the numbers of true positives, false positives and false negatives respectively for class c. From these, we calculate recall (Re_c), precision (Pr_c) and F measure (F_c) for class c [2]. We also calculate overall recall (η_μ). We evaluate the results using 5-fold cross validations. The average recall, precision and F measures have been reported in Table VII. Notice that the F measure for all the classes are above 0.9. We further find that overall recall $\eta_\mu = 0.94$. Next we evaluate the repeatability of the proposed method. For this, we divide our entire dataset in 5 equal subsets (SS1-SS5) randomly. The F measures for each class in different subsets are reported in Table VIII. We find that for each class, the mean value of F measure (over all the subsets) is high while the value of standard deviation is insignificant. The F measures across different subsets of data are thus consistent and stable. We further calculate overall F measure across the classes for all the subsets (SS1-SS5). In any subset, we first calculate overall true positive across the classes, given by $\text{TP} = \sum_c \text{TP}_c$. Similarly, we calculate overall false positive ($\text{FP} = \sum_c \text{FP}_c$) and overall false negative ($\text{FN} = \sum_c \text{FN}_c$). We calculate overall F measure using these values (see Table VIII). The overall F measure is found to be consistent across the subsets.

We also evaluate the the value of overall recall (η_μ) for different subsets. The results are reported in Table IX. Notice that η_μ has a high mean value across subsets with a low standard deviation. This shows that the η_μ in the proposed method has low dependence on the choice of data.

Next, we observe the effect of adding trees as redundant features are removed at each pass. For this, we calculate overall F Measure at each construction pass of the proposed method until convergence is reached. From Fig. 5, we find that as more trees are added and redundant features are removed following Algorithm 1, the overall F Measure improves. This experimentally justifies the proposed method. Further, notice from Fig. 5 that overall F measure also almost saturates at convergence (after the 15^{th} pass). This is expected since overall

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

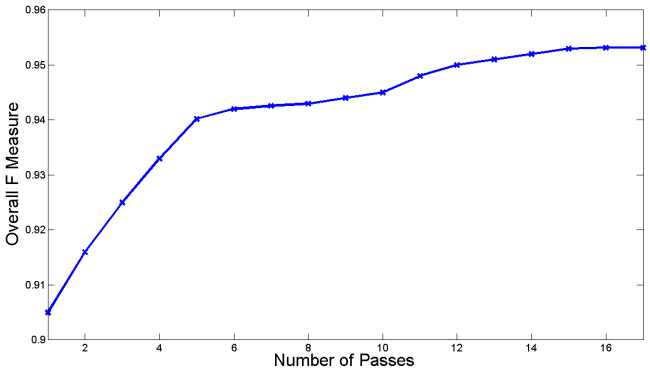


Figure 5: Change of overall F Measure as convergence is achieved through a number of passes in the proposed IRF.

recall saturates at convergence. Thus, saturation of η_μ is also validated from Fig. 5.

3) Comparisons: We compare the classification performance of the proposed method with other state-of-the-art statistical classifiers. Note that, typically the converged forest in the proposed method contains ∼80 trees. So, for the competing tree based method, we take 80 trees as well. Similar to the previous datasets, we compare our method with AdaBoost.M2 (ADA), RUSBoost (RUS), random forest (RF) [9] and support vector machine (SVM) with RBF kernel [33]. Since ours is a three class problem, we use one versus one [36] comparison strategy for SVM to report the results. The parameters of SVM gamma and cost are set through cross validations. In particular, we take the value of gamma to be 0.9 and cost to be 10000.

For each of the above methods, we calculate recall (Re), precision (Pr) and F measure for each of three classes (ferrite, bainite and martensite) through 5-fold cross validations. For bainite, martensite and ferrite classes, the results can be visualized in Fig. 6. From this figure, notice that the proposed IRF method outperforms its state-of-the-art competitors by significant margins in terms of all the performance measures. Results in some candidate regions have been presented in Fig. 7. The results in Fig. 7 show that the proposed method performs better for all the classes compared to the competing methods.

Finally, we focus on the overall F measure and overall recall (η_μ). The comparative results for these two quantities is presented in Fig. 8. Observe that our method has a greater value of overall F Measure and η_μ. The above discussions and figures clearly establishes the superiority of the proposed improved random Forest (IRF) compared to other states-of-the-art methods for classifying steel phases from *nital* etched images of DP steel. Using unoptimized MATLAB code, the training time in the present problem is ∼2 hour and the testing time (for classification) in a typical image with ∼0.1 million pixels is nearly 5 minutes using a computer with Intel Xeon processor and 16 GB of memory. The manual classification takes typically around 3-4 days. Thus the proposed method is significantly faster than the manual process. The time efficiency of our method can be improved even further by code optimization.

D. Advantages Compared to Conventional Random Forest

IRF is a modification of conventional random forest. It has already been observed that given the same number of trees, IRF outperforms RF for benchmark, breast cancer and steel microstructure datasets. Next we investigate if increasing the number of trees in RF can lead to results comparable to IRF. IRF required <100 trees for all the datasets we use. So, we start with an RF of 100 trees and increase the number of trees up to 2000 (in step of 100). For each dataset, we find the numbers of trees in RF that produce the lowest average error (see Table X). The best results in terms of average error of RF and of IRF have been presented in Table X. Notice that even with much larger number of trees, RF does not beat the proposed IRF. Thus we find that our method beats RF with less computational burden.

IV. Conclusions

We propose a fast and accurate solution for automatic classification by improvising random forest classifier. The proposed classifier not only removes redundant features, but also dynamically change the size of the forest (number of trees) to produce optimal performance in terms of classification accuracy. The proposed method out-performs state-of-the-art competitors in benchmark datasets. Our classifier has also proven to be useful in classification problems of mitosis detection from breast tissues. The proposed method has shown promising results in the problem of phase classification from dual phase steel microstructures. The proposed classifier has the potential to be applied in industrial applications. In future, we would explore random forest guided autoencoder for feature encoding.

Appendix A
On Average Number of Nodes

In our formulation, a node splits only if there are data from more than one classes in that node. Let the bootstrap sample for growing a random tree has s number of feature vectors. So, in each tree of the forest, there could be at most s leaf nodes. Since each tree in our algorithm is a complete binary tree, a tree can have maximum $(2s-1)$ nodes. Number of non-leaf nodes in the tree is $(s-1)$. So, a total of $(2s-1)$ nodes are generated when $(s-1)$ nodes split. In other words, probability of a tree having $(2s-1)$ nodes is equal to the probability that $(s-1)$ nodes split. Since the bootstrap sample is chosen with replacement, a node may have several instances of a particular data. Let there be total $\mathbf{C}$ classes in our dataset. In a node with d number of data points, the probability that all the data are from the same class is $\left(\frac{1}{\mathbf{C}}\right)^{d-1}$. Also, a node may split only if there are more than one data points$(d \geq 2)$ in the node. So, putting $d = 2$, the probability of node splitting (χ) is: $\chi \leq 1 - \left(\frac{1}{\mathbf{C}}\right)$. Consequently, the probability of successive splitting of $s-1$ number of nodes is $\leq \chi^{(s-1)}$. So, the maximum probability of having $2s-1$ nodes in a tree is $\chi^{(s-1)}$. Thus, average number of nodes in a tree is:

$$N_{\text{av}} \leq \sum_{k=1}^{s} (2k-1)\,\chi^{(k-1)}. \tag{21}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

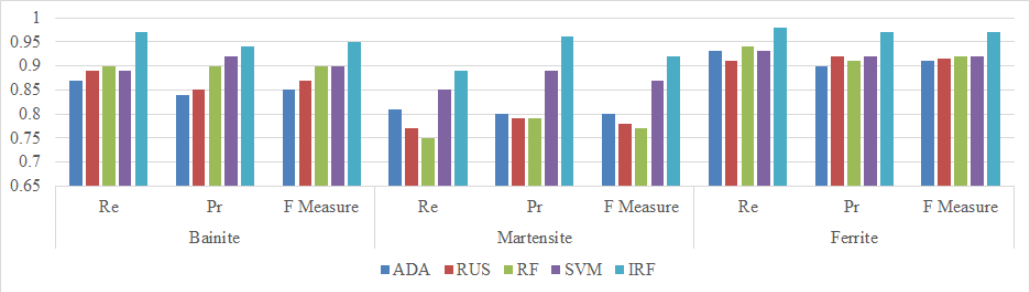


Figure 6: Comparative performances of different methods in terms of recall (Re), precision (Pr) and F Measure for different phases of steel microstructure.

Table X: Number of trees (with average error) in RF and IRF for lowest average error for benchmark datasets

Dataset	MNIST	MNIST-Rot	MNIST-Bkg	MNIST-RandB	MNIST-RotB
RF	1700 (16.9)	1200(3.3)	1300 (6.2)	1500 (16.9)	1200 (5.1)
IRF	80 (13.7)	60 (3.9)	65 (5.1)	70 (16.7)	60 (4.9)
Dataset	USPS	ISOLET	G50c	Letter	Madelon
RF	1400 (5.9)	1000 (6.1)	900 (18.9)	1100 (4.2)	1600 (13.9)
IRF	60 (5.5)	85 (5.7)	45 (18.7)	75 (3.7)	65 (6.3)

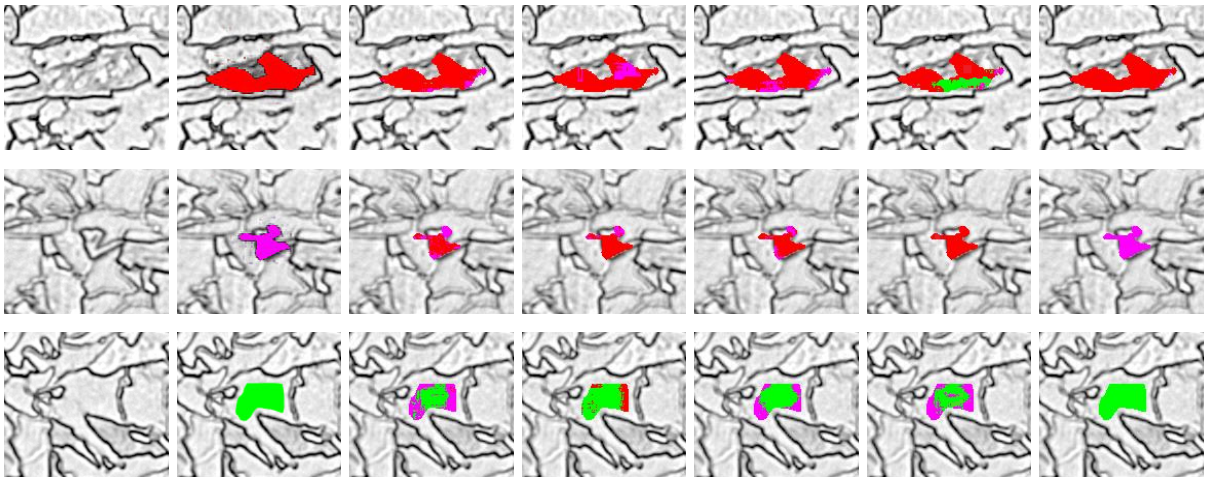


Figure 7: Results using different methods in some candidate regions of DP steel micrographs. Column 1: original region, column 2: ground truth (red: bainite, magenta: martensite, green: ferrite), classification results using - column 3: AdaBoost, column 4: RUSBoost, column 5: RF, column 6: SVM and column 7: proposed IRF.

Equation (21) clearly shows that N_{av} is not a function of number of trees (B). Hence, the change in average number of nodes per tree (N_{av}) is insignificant as the number of trees in the forest is increased. Since $\chi < 1$, for sufficiently large s (as in our case), limiting value of N_{av} is approximated as:

$$N_{\text{av}} \approx \frac{\chi + 1}{(1-\chi)^2}. \tag{22}$$

Appendix B
On Generic Nature of the Proposed Formulation for Random Forest Performance

Conventional random forest proposed by Breiman [9] contains no feature reduction. So, according to our formulation, $\Delta v = 0$ in (18). Also, conventional random forest do not find important features in an iterative manner. So, $\Delta u = 0$. Then, for random forest proposed in [9], (18) can be re-written as:

$$d\eta \approx \lambda \nu \Delta B. \tag{23}$$

According to [9], the classification error converges for a large number of trees. From (15), notice that $(1-\rho) < 1$ and $\left(1 - q^{N_{\text{av}}}\right) < 1$. If B is very large, then we get $\nu \to 0$. As a result $d\eta \approx 0$. Thus classification accuracy (and hence classification error) converges for large number of trees. In this way, our model can explain the convergence of error in random forest proposed in [9].

Next we show that our definition of strength and correlation satisfies the properties proposed in [9]. According to [9], both strength and correlation increases when f is increased. From (9), we have strength $\eta_s = 1 - \left(1 - q^{N_{\text{av}}}\right)^B$. We get:

$$\frac{\partial \eta_s}{\partial f} = B N_{\text{av}} q^{N_{\text{av}}-1} \left(1 - q^{N_{\text{av}}}\right)^{B-1} \frac{\partial q}{\partial f}. \tag{24}$$

Now, from (7), it can be shown that, when $v \geq f$:

$$\frac{\partial q}{\partial f} \approx \frac{\Delta q}{\Delta f} = \frac{u\ v!\,(u+v-f)\,!}{(u+v)\,!\,(v-f+1)\,!} > 0. \tag{25}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2018.2834830, IEEE Transactions on Image Processing

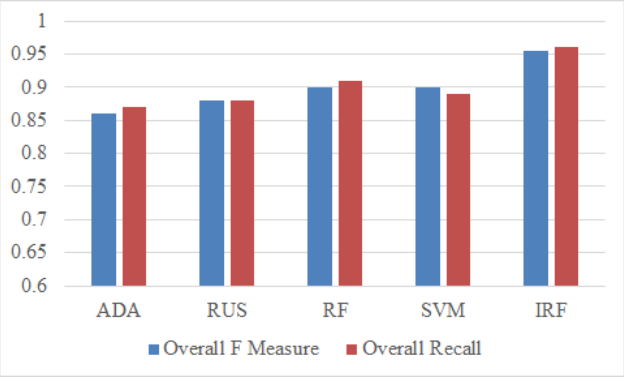


Figure 8: Comparative performances of different methods in terms of overall F Measures and overall recall (η_μ).

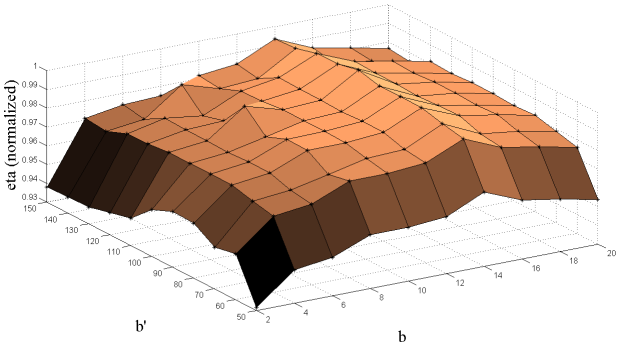


Figure 9: Overall recall (η_μ) (normalized) w.r.t. b and b'.

So, from (24), we get $\frac{\partial \eta_s}{\partial f} > 0$. Next, from (12) and (11):

$$\frac{\partial \eta_c}{\partial f} = -\frac{\partial \left(1-\rho\right)^{\frac{B}{2}}}{\partial f} = \frac{B}{2}\left(1-\rho\right)^{\frac{B}{2}-1}\frac{\partial \rho}{\partial f}$$
$$= \frac{BN_{\text{av}}\rho'^{N_{\text{av}}-1}}{2}\left(1-\rho\right)^{\frac{B}{2}-1}\frac{\partial \rho'}{\partial f}. \quad (26)$$

From (11), it can be shown that:

$$\frac{\partial \rho'}{\partial f} \approx \frac{\Delta \rho'}{\Delta f} = \frac{((u+v-f)!)^2}{(u+v-2f)!(u+v)!}\times$$
$$\left[\frac{(u+v-f+1)^2}{(u+v-2f+2)(u+v-2f+1)} - 1\right]. \quad (27)$$

Notice that $\frac{\Delta \rho'}{\Delta f} > 0$ for $f \geq 1$. So, $\frac{\partial \eta_c}{\partial f} > 0,\ \forall f \geq 1$. Thus correlation increases as f increases. Hence, we conclude that our model is a generic model for random forest.

Appendix C
Choices of b and b'

b is the number of bins in the histogram of contour lines. The mean-variance histogram also has a dimension of $b \times b$. The entropy histogram on the other hand is b' dimensional. Thus b and b' are two crucial parameters in feature extraction. In order to find suitable values of b and b', we take a particular (b, b') tuple and evaluate overall recall (η_μ) after 5-fold cross validations. In this way, η_μ is evaluated for a range of b and b'. Then we normalize η_μ (w.r.t. the maximum value of η_μ for different b and b') and make a 3D plot of b, b' and normalized η_μ (see Fig. 9). It can be observed from Fig. 9 that the peak of the plot (that indicates maximum value of η_μ) occurs at $b = 14,\ b' = 110$. So, we choose $b = 14,$ and $b' = 110$ for our experiments.

Appendix D
On Initial Number of Trees (B_0)

Let N_τ be the number of nodes in tree τ of the initial forest of B_0 trees. So we get average number of nodes per tree: $N_{\text{av}} = \left(\sum_{\tau=1}^{B_0} N_\tau\right)/B_0$. Now, when we add one more tree to the forest, the average number of nodes per tree:

$$N'_{\text{av}} = \frac{\sum_{\tau=1}^{B_0+1} N_\tau}{B_0+1} = \frac{N_{\text{av}}B_0 + N_{(B_0+1)}}{B_0+1}, \quad (28)$$

where $N_{(B_0+1)}$ is the number of nodes in tree (B_0+1). Let $N_{B_0+1} = N_{\text{av}} + \epsilon$, where ϵ is an integer. Then form (28):

$$N'_{\text{av}} = N_{\text{av}} + \frac{\epsilon}{B_0+1}. \quad (29)$$

Now, in order to keep average number of nodes per tree (N_{av}) constant (according to Proposition 1), we must have $N'_{\text{av}} = N_{\text{av}}$. This is possible only when $\frac{\epsilon}{B_0+1} < 1$. Consequently, $B_0 > \epsilon - 1$. Note that ϵ is the amount of variation in the number of nodes from the average number of nodes. We can approximate ϵ as the standard deviation in the number of nodes per tree. Hence, from Appendix A, we can write:

$$\epsilon \approx \sqrt{\sum_{k=1}^{s}\left(2k-1-N_{\text{av}}\right)^2 \chi^{(k-1)}}. \quad (30)$$

Since $\chi < 1$, we neglect higher order (> 2) terms of χ. So:

$$\epsilon \approx \sqrt{\frac{\left(N_{\text{av}}\left(\chi-1\right)\right)^2 - 2N_{\text{av}}\left(1-\chi^2\right) + \chi^2 + 6\chi + 1}{\left(1-\chi\right)^3}}. \quad (31)$$

From this, we can determine the initial number of trees.

References

[1] C. Luo, Z. Wang, S. Wang, J. Zhang, and J. Yu, "Locating facial landmarks using probabilistic random forest," *Signal Processing Letters, IEEE*, vol. 22, no. 12, pp. 2324–2328, Dec 2015.

[2] A. Paul and D. Mukherjee, "Mitosis detection for invasive breast cancer grading in histopathological images," *Image Processing, IEEE Transactions on*, vol. 24, no. 11, pp. 4041–4054, Nov 2015.

[3] A. Criminisi and J. Shotton, *Decision forests for computer vision and medical image analysis*. Springer Science & Business Media, 2013.

[4] T. M. Khoshgoftaar, M. Golawala, and J. V. Hulse, "An empirical study of learning from imbalanced data using random forest," in *ICTAI 2007*, vol. 2. IEEE, 2007, pp. 310–317.

[5] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 39, no. 2, pp. 539–550, 2009.

[6] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple classifier systems*. Springer, 2000, pp. 1–15.

[7] N. Quadrianto and Z. Ghahramani, "A very simple safe-bayesian random forest," *PAMI, IEEE Trans. on*, vol. 37, no. 6, pp. 1297–1303, June 2015.

[8] A. Paul, A. Dey, D. P. Mukherjee, J. Sivaswamy, and V. Tourani, "Regenerative random forest with automatic feature selection to detect mitosis in histopathological breast cancer images," in *MICCAI 2015*. Springer, 2015, pp. 94–102.

[9] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[10] T. M. Oshiro, P. S. Perez, and J. A. Baranauskas, "How many trees in a random forest?" in *MLDM*. Springer, 2012, pp. 154–168.

[11] P. Latinne, O. Debeir, and C. Decaestecker, "Limiting the number of trees in random forests," in *Multiple Classifier Systems*. Springer, 2001, pp. 178–187.

[12] A. Cuzzocrea, S. L. Francis, and M. M. Gaber, "An information-theoretic approach for setting the optimal number of decision trees in random forests," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1013–1019.

[13] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn, "Bias in random forest variable importance measures: Illustrations, sources and a solution," *BMC bioinformatics*, vol. 8, no. 1, p. 25, 2007.

[14] A. Montillo, J. Shotton, J. Winn, J. E. Iglesias, D. Metaxas, and A. Criminisi, "Entangled decision forests and their application for semantic segmentation of ct images," in *Biennial International Conference on Information Processing in Medical Imaging*. Springer, 2011, pp. 184–196.

[15] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[16] A. Gogna and A. Majumdar, "Semi supervised autoencoder," in *International Conference on Neural Information Processing*. Springer, 2016, pp. 82–89.

[17] M. Lichman, "Uci machine learning repository," 2013.

[18] J. J. Hull, "A database for handwritten text recognition research," *PAMI, IEEE Transactions on*, vol. 16, no. 5, pp. 550–554, 1994.

[19] O. Chapelle *et al.*, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]," *IEEE Trans. on Neural Networks*, vol. 20, no. 3, pp. 542–542, 2009.

[20] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.

[21] I. Guyon, S. R. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the nips 2003 feature selection challenge." in *NIPS*, vol. 4, 2004, pp. 545–552.

[22] Http://ludo17.free.fr/mitos_2012/dataset.html. Available as on 14.03.2017. [Online]. Available: http://ludo17.free.fr/mitos_2012/dataset.html

[23] Http://amida13.isi.uu.nl/?q=node/3. Available as on 14.03.2017. [Online]. Available: http://amida13.isi.uu.nl/?q=node/3

[24] Https://mitos-atypia-14.grand-challenge.org/dataset/. Available as on 14.03.2017. [Online]. Available: https://mitos-atypia-14.grand-challenge.org/dataset/

[25] Q. Lai *at al.*, "Damage and fracture of dual-phase steels: Influence of martensite volume fraction," *Materials Science and Engineering: A*, vol. 646, pp. 322–331, 2015.

[26] R. Rocha *at al.*, "Microstructural evolution at the initial stages of continuous annealing of cold rolled dual-phase steel," *Materials Science and Engineering: A*, vol. 391, no. 1, pp. 296–304, 2005.

[27] M. R. Plichta *et al.*, "Chemical polishing of steel for optical microscopy," *Metallography*, vol. 9, no. 5, pp. 455–457, 1976.

[28] T. Bylander, "Estimating generalization error on two-class datasets using out-of-bag estimates," *Machine Learning*, vol. 48, no. 1-3, pp. 287–297, 2002.

[29] H. Ishwaran, "The effect of splitting on random forests," *Machine Learning*, vol. 99, no. 1, pp. 75–118, 2014.

[30] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[31] Y. Freund, R. E. Schapire *et al.*, "Experiments with a new boosting algorithm," in *ICML*, vol. 96, 1996, pp. 148–156.

[32] C. Seiffert *et al.*, "Rusboost: A hybrid approach to alleviating class imbalance," *SMC, Part A: Systems and Humans, IEEE Trans. on*, vol. 40, no. 1, pp. 185–197, 2010.

[33] B. Scholkopf *et al.*, "Comparing support vector machines with gaussian kernels to radial basis function classifiers," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2758–2765, 1997.

[34] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *International Conference on Medical Image Computing and Computer-assisted Intervention*. Springer, 2013, pp. 411–418.

[35] C. D. Malon and E. Cosatto, "Classification of mitotic figures with convolutional neural networks and seeded blob features," *Journal of pathology informatics*, vol. 4, 2013.

[36] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *Neural Networks, IEEE Transactions on*, vol. 13, no. 2, pp. 415–425, 2002.