Comparing Transfer Learning to Feature Optimization in Microstructure Classification

Debanshu Banerjee[1] and Taylor D. Sparks*[2]

[1]Metallurgical and Material Engineering Department, Jadavpur University
[2]Department of Materials Science and Engineering, The University of Utah

August 9, 2021

Abstract

Human analysis of research data is slow and inefficient. In recent years machine learning tools have advanced our capability to perform tasks normally carried out by humans, such as image segmentation and classification. In this work, we seek to further improve binary classification models for high throughput identification of different microstructural morphologies. We utilize a dataset with limited observations (133 dendritic structures, 444 non-dendritic) and employ data augmentation via rotation and translation to enhance the dataset six-fold. Then, transfer learning is carried out using pre-trained networks VGG16, InceptionV3, and Xception achieving only moderate F1 scores (0.801 to 0.822). We hypothesize that feature engineering could yield better results than transfer learning alone. To test this, we employ a new nature-inspired feature optimization algorithm, the Binary Red Deer Algorithm (BRDA), to carry out binary classification and observe F1 scores above 0.93.

1 Introduction

Materials science is centered around the concept of understanding, extracting, and exploiting relationships between structure, property, processing, and characterization of materials. Images of a material's crystal or microstructure can be invaluable in establishing structure-processing-property linkages. However, the human eye itself can only resolve objects as small as $\sim 50\,\mu$m. Investigating smaller details than this requires microscopy like optical microscopy, transmission electron microscopy (TEM), scanning electron microscopy (SEM),

atomic force microscopy (AFM) and others. Each of these techniques offers different resolution limitations along with other advantages and disadvantages.

Analysis of microstructure images can yield an enormous amount of information about a material! For example, some alloys exhibit a dendritic structure characterized by the presence of myriad snow-flake like dendrites that form during the solidification process of casting molten metals. Dendrites can form when the interface of the solid cast and the liquid alloy has a lower temperature as compared to the remaining melt resulting in a temperature gradient known as constitutional undercooling [1]. Since dendrites can modify the mechanical properties in beneficial or detrimental ways, [2] it is important to classify microstructures of materials as dendritic and non-dendritic.

Traditionally, humans have leveraged training and extensive domain knowledge to interpret and categorize microscopy images. This poses challenges for interdisciplinary research as well as automated, human-out-of-the-loop experimentation. Moreover, even highly trained humans are still prone to errors and bias during materials characterisation. This gave rise to the implementation of computational methods on the study of dendritic microstructures which started back in 1998 to focus on the evolution of the said microstructures using phase field modelling [3]. Later attempts have been made on three dimensional reconstruction of microstructures captured from different microscopes which has helped in getting better insights about the mechanical properties [4, 5]. But all these works were simulation based and time-consuming.

An alternative to physics-based models are statistical, data-driven models that trade accuracy for speed by leveraging correlations and patterns in data. Machine learning models have been used extensively to analyze various microstructural morphologies over the last two decades[6]. One such example is the development of optimal morphology derivation from a given microstructure using Bayesian optimization and kinetic Monte Carlo simulation[7]. Transfer learning has been deployed for microstructure reconstruction and structure property predictions [8] while support vector regression and multi-layer perceptron (MLP) is used to predict information related to dendrite formation in order to improve high-temperature creep and fatigue resistance [9]. Different types of industry relevant titanium alloy microstructures can now be classified using convolution neural networks [10]. Particle swarm optimization algorithm has been deployed to classify the casting techniques resulting in dendritic and non-dendritic microstructures in aluminium metal matrix composites [11]. Advanced computer vision techniques have even been utilized in order to generalize microstructure morphology classification beyond individual alloy systems[12, 13, 14].

In this work, we turn to a recently reported, nature-inspired algorithm called Binary Red Deer Algorithm (BRDA) [15] to implement feature engineering to classify microstructural morphologies as dendritic or non-dendritic. For reference we also present a comparative study between transfer learning approach and the BRDA feature engineering approach in classifying microstructural morphologies. We show that BRDA outperforms advanced transfer learning techniques and has promise in the evolving field of computer vision in materials

informatics.

2 Methods

An image dataset consisting of two different types of micrographs, dendritic and non-dendritic, has been considered in this work. The number of images in the dendritic class is 133 while that in the non-dendritic class is 444 which are quite low from a machine learning standpoint. Consequently, we have performed a data augmentation on this dataset by translating and rotating the images at various angles which has given us a six-fold increase in the number of images per class. The number of images after data augmentation in the dendritic and non-dendritic class are 826 and 2568, respectively. Following this, we have used transfer learning and feature engineering approaches as elaborated in subsection 2.2 and subsection 2.3 respectively to develop binary classification models on the augmented dataset.

2.1 Dataset

The dataset originates from the Dissemination of Information Technology for Promotion of Materials Science (DoITPoMS), a web-based initiative which started in the Materials Science and Metallurgy Department at Cambridge University [16]. This comprises a collection of micrographs covering a wide range of specimen types like ceramic, metal or alloy, device, composite, polymer, foam, etc. and microscopy techniques like optical micrography, SEM or TEM. Information related to chemical composition and processing technique is also available as metadata for every microstructure. Additionally, in some cases microstructures corresponding to a particular specimen are available at different magnifications. In this work, we have considered microstructures from 21 different alloy systems including both dendritic and non-dendritic classes. Figure 1 shows some example images used to build the proposed binary classification models.

2.2 Transfer Learning Approach

Three different pre-trained CNN architectures; `VGG-16` [17], `InceptionV3` [18] and `Xception` [19] have been used. These networks have been trained and validated separately using the augmented dataset. Transfer learning approach has been implemented to train the classification models. Initially, all the layers were frozen and the weights corresponding to the `ImageNet` dataset [20] were used to validate the networks on the augmented microstructural dataset. Subsequently, a number of different hyperparameter combinations were tried by freezing and unfreezing different layers. For frozen layers, weights corresponding to training on the `ImageNet` dataset were used. Figure 2 and Figure 3 show the architectures along with the frozen and unfrozen layers for `VGG16` and `InceptionV3` networks, respectively, that have given the maximum validation

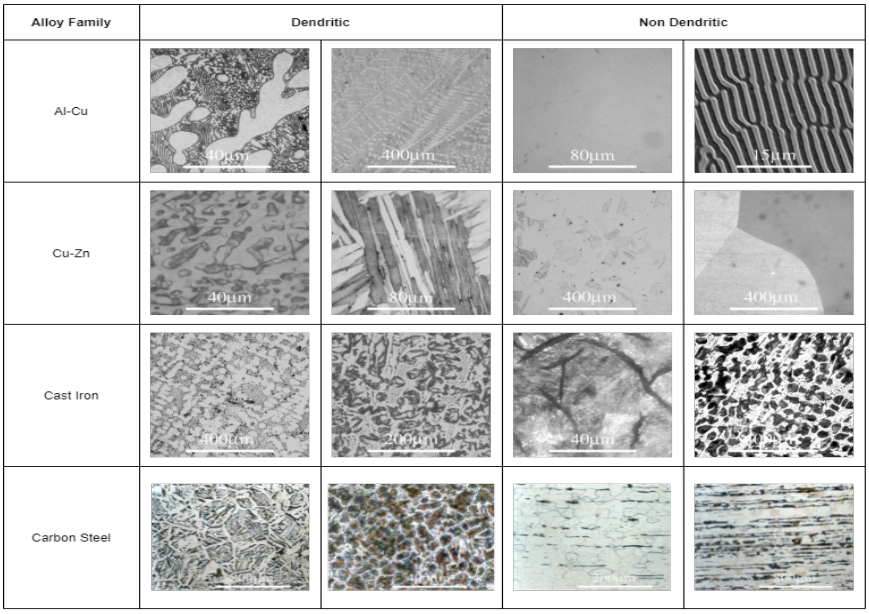

Figure 1: Example microstructures considered in this work.

accuracy on the augmented dataset where grey boxes indicate the frozen layers. In the case of `VGG16`, the first four convolution layers have been frozen. This is followed by three repeating units of three convolution layers separated by a maxpool layer. For each of these three units, a general pattern of freezing the middle convolution layer has been observed. The remaining two fully connected layers and softmax layers have been trained on the augmented dataset. In `InceptionV3`, the first five convolution layers have been frozen. The layers at the end in the form of fully connected and softmax have been trained with respected to our dataset. In between, we have three different types of inception modules (L_1, L_2 and L_3) where convolution layers have been frozen at random based on a trial and error approach. In the case of `Xception` transfer learning did not prove to be efficient and all the layers were trained on the augmented dataset. The validation accuracies for `VGG16`, `InceptionV3` and `Xception` considering the discussed architectures are 0.8313, 0.8688 and 0.8594 respectively.

2.3 Feature Engineering Approach

Since `InceptionV3` has given the maximum validation accuracy among the three pre-trained networks, it is used as a feature extractor on the augmented dataset. This gives 12,289 features per image. This feature set is used to train two machine learning classifiers, random forest (RF) [21] and MLP [22]. The classification accuracies for RF and MLP are 0.4198 and 0.5556 respectively. In

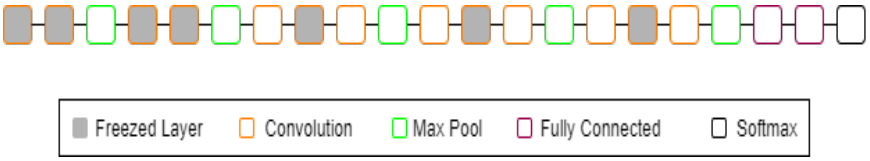

Figure 2: Architecture of `VGG16` network depicting the frozen and unfrozen layers

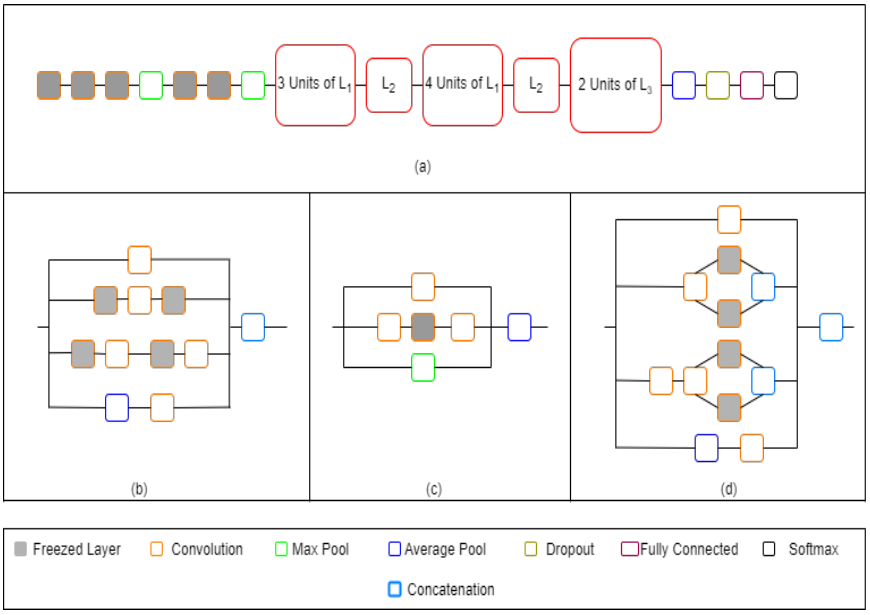

Figure 3: Schematic diagram showing (a) Architecture of `Inception3` network depicting the frozen and unfrozen layers, (b) `Inception` module 1 (L_1), (c) `Inception` module 2 (L_2) and (d) `Inception` module 3 (L_3)

order to improve upon the classification results, a meta-heuristic optimization algorithm, BRDA, has been deployed. BRDA reduces the size of the feature vector to 6072 features per image. Following this feature selection, RF and MLP have been trained on the optimized feature set and the classification accuracies are 0.9735 and 0.9855 for RF and MLP, respectively. BRDA has been delineated in subsection 2.4.

2.4 Binary Red Deer Algorithm (BRDA)

Red Deer Algorithm (RDA) is a recently proposed nature inspired meta-heuristic optimization algorithm [23] which is derived from the mating behavior of a sub-species of red deer known as Scottish red deer. During a breeding season, the male red deer (RD) begin the mating ritual by roaring which attracts the female counterparts called hinds. These male RDs' are categorized as commanders, ones having higher roaring intensity, and the rest are called stags. Every commander forms a harem, a group of hinds that mate with that particular commander. Harem size depends on the power of the commander defined by its fitness value. Besides, a commander also has the ability to mate with hinds belonging to other harems. Stags, on the other hand, mate randomly with their nearest hinds. This phenomenon of mating ensures a competitive evolution at each stage of the algorithm which explores the entire space of RDs'. In this work, we aim to use a binarized form of RDA called Binary Red Deer Algorithm (BRDA) for feature selection in order to choose an optimized subset of features from the whole set of features obtained from the augmented dataset using InceptionV3. The aim here is to maximize the classification accuracy simultaneously minimizing the number of features. Therefore, in this work, feature selection is modeled as a binary optimization problem, where the solutions are limited to {0,1}. In the BRDA, we first randomly initialize a vector of real numbers called RD of size m, the total number of features in the feature set.

$$RD = [X_1, X_2, X_3, ..., X_m] \tag{1}$$

RD is converted into a binary vector (BRD) comprising only 0 and 1 using the Sigmoid function shown in Equation 2. Here 1 indicates that the corresponding feature is selected in the feature subset and vice versa for 0. The real values of RD are converted into binary values using a threshold of 0.5 as expressed in Equation 3.

$$S(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

$$X_i = \begin{cases} 1 & \text{if } S(X_i) > 0.5 \\ 0 & \text{if } S(X_i) \leq 0.5 \end{cases} \tag{3}$$

where $i \in [1, m]$. The quality of the RDs at every iteration of the algorithm is evaluated by a fitness function as expressed in Equation 12.

2.4.1 Initialization of the RD population

At first, a RD population of size N is initialized randomly. Based on fitness values, the top RDs represent the males (N_1), and the rest of the RDs represent the hinds (N_2). The fraction of the RD to be considered as male is a hyperparameter for BRDA and needs to be specified manually.

2.4.2 Roaring of male RDs

In order to successfully roar and attract hinds, the male RDs may change their positions according to Equation 4. If the fitness value of the male RDs at the new position is better than that in the original position, then the position of the RD is updated and roaring is considered to have been successful. Otherwise, the old position is retained.

$$new = \begin{cases} old + a_1 \times ((upper - lower) \times a_2 + lower) & if\ a_3 \geq 0.5 \\ old - a_1 \times ((upper - lower) \times a_2 + lower) & if\ a_3 < 0.5 \end{cases} \tag{4}$$

Here, *old* is the original position of the male RD whereas *new* is the position to which a male RD moves during the roaring procedure. a_1, a_2 and a_3 are randomly generated numbers from a uniform distribution of 0 and 1 while $upper = 1$ and $lower = -1$ are the upper and lower bounds of the search space of the entire RD population.

2.4.3 Distinguishing Commanders from Stags

Among the male RDs, the top N_3 are selected as commanders according to Equation 5

$$N_3 = floor\,(\gamma \times N_1) \tag{5}$$

where $\gamma \in [0, 1]$ is a hyperparameter which is to be specified manually. Each of these commanders competes with the $N_1 - N_3$ number of stags randomly and two new solutions, New_1 and New_2 are generated as expressed in Equation 6 and Equation 7 respectively. The position of the commander is updated using the solution which results in the best fitness value among the commander, the stag, and the two new solutions.

$$New1 = \frac{Commander + Stag}{2} + (b_1 \times ((upper - lower) \times b_2) + lower) \tag{6}$$

$$New2 = \frac{Commander + Stag}{2} - (b_1 \times ((upper - lower) \times b_2) + lower) \tag{7}$$

$Commander + Stag$ represents addition of the two vectors corresponding to commander and stag respectively. b_1 and b_2 are generated using a uniform distribution function in [0,1].

2.4.4 Formation of Harems

Since this is a minimization problem, better quality of solution is determined by a lower fitness value. Therefore, we find the power of each commander according to Equation 8

$$P_j = F - f_j \tag{8}$$

where P_j and f_j are the power and fitness value of the jth commander respectively and $j \in [1, N_3]$. F is the sum of fitness values of all commanders.

Equation 9 represents the fraction of the total number of hinds that form a harem with a particular commander.

$$N_{4_j} = floor\left\{P_j \times N_2\right\} \tag{9}$$

where N_{4_j} is the number of hinds that belongs to the jth harem.

2.4.5 Mating of Commanders

In each harem, all hinds mate with the respective commander to produce offspring according to Equation 10.

$$offspring = \frac{Commander + Hind}{2} + (upper - lower) \times c \tag{10}$$

Here c is a randomly generated number between 0 and 1. Besides, all the commanders can mate with hinds from all other harems. Consequently, a new population of RDs are generated which are stored in *offspring pool*

2.4.6 Mating of Stags

Each stag mates with its nearest hind irrespective which of harem the hind belongs to. The distance between a stag and all hinds is calculated by Equation 11.

$$d_k = \sqrt{\sum_{k \in N_2} \left(stag_k - hind_k^j\right)^2} \tag{11}$$

where d_k is the distance between a stag and the jth hind. The hind at the minimum distance is selected for mating, which takes place according to Equation 10, with the stag replacing the male commander. The offspring formed in this process are added to the *offspring pool*.

2.4.7 Selection of Next Generation

After the mating process is completed, the offspring from the *offspring pool* are shuffled with the original population. The top N RDs are selected according to fitness values as the next generation and the rest of the solutions are discarded.

2.4.8 Terminating the BRDA

The process is stopped when *iter* number of iterations are completed. The RD with the best fitness value in the final generation represents the optimized feature subset which is used to train the RF and MLP classifiers.

2.4.9 Fitness Evaluation

Fitness quantifies the quality of BRDA solution. A learning algorithm based on RF classifier is used to evaluate the performance of a particular feature subset along with the whole feature set. The fitness function consists of two components: classification accuracy and the number of features. Our objective is to achieve the highest classification accuracy minimizing the number of features. A higher classification accuracy and fewer features implies a low fitness value. The fitness function is shown in Equation 12.

$$Fitness = \kappa \times \frac{|Selected|}{|Total|} + (1 - \kappa) \times \psi \tag{12}$$

where $|Selected|$ is the number of features in the selected feature subset, $|Total|$ is the total number of features of the dataset, ψ is the classification error of the feature subset, and $\kappa \in [0, 1]$ indicates the relative weight assigned to the number of features and the classification error.

The time complexity of BRDA is expressed as $O(iter * N^2 * (t + m))$ where t is the time complexity in calculating the fitness of a particular RD using the RF classifier.

3 Results and Discussion

The code to build the transfer learning and feature engineering based models have been developed predominantly by using two Python libraries, `TensorFlow Core v2.5.0` and `Scikit-Learn 0.24.2`. Early stopping has been deployed in case of training the pre-trained networks to minimize the number of epochs and obtain the optimum results [24].

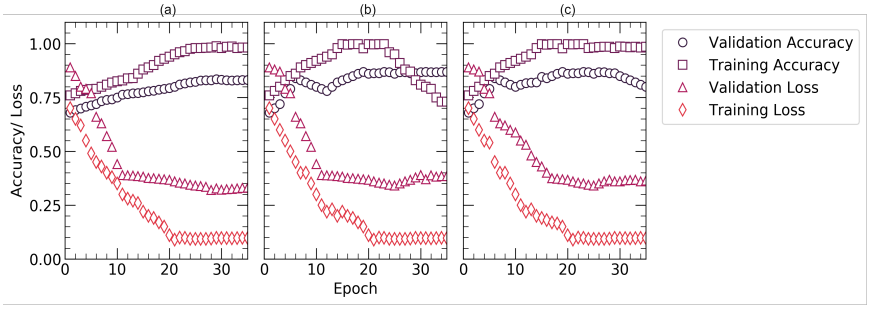

Figure 4: Plots showing the performance of (a) `VGG16`, (b) `InceptionV3` and (c) `Xception` networks on the augmented dataset

Algorithm 1 Algorithm of BRDA

Input: *iter*, N

Output: Optimized feature set $X' = (x_1, x_2, ..., x_m)$

```
Initialize RD population
Determine N_1 males and N_2 hinds based on fitness of RDs
for e_1 ← 1 to iter do
    for e_2 ← 1 to N_1 do
        Update position of RD if fitness increases
    end for
    Level stags and commanders
    for e_2 ← 1 to N_3 do
        Competition between commander and stag
        Update position of the commander
    end for
    Creation of harems
    for e_2 ← 1 to N_3 do
        Mating commander with hinds from its harem
        Mating commander with hinds from other harems
    end for
    for e_2 ← 1 to (N_1 − N_3) do
        Select the closest hind from the nearest harem
        Mating of stag with the selected hind
    end for
    Selection of the next generation from offspring pool
    Update X' if a better solution is encountered
end for
return X'
```

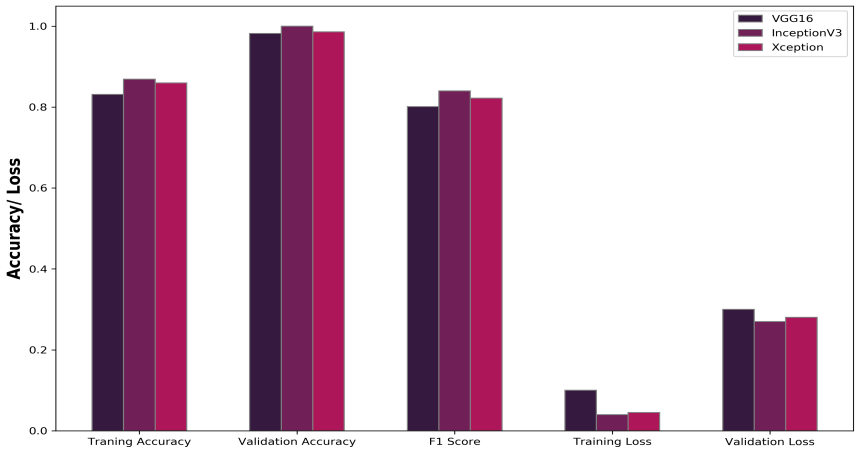

Figure 5: Comparison between the accuracies and losses for training and validation of the final pre-trained network architectures

Consequently, `VGG16`, `Inception3` and `Xception` were stopped after 35, 37, and 43 epochs, respectively as the ADAM optimizer [25] failed to reach any maxima beyond this point. Figure 4 delineates the epoch wise training accuracies, validation accuracies, training errors and validation errors for each of the three networks. Early stopping gives us validation accuracies of 0.8313, 0.8688 and 0.8594 and validation losses of 0.3, 0.27 and 0.28 for VGG16, Inception3 and Xception respectively and the corresponding training accuracies for these networks are 0.9819, 0.9997 and 0.9856 while the training losses are 0.1, 0.04 and 0.045 respectively as shown in Figure 5. These results indicate that even the best combination and frozen and unfrozen layers fail to provide a generalized binary classifier due to overfitting. Therefore, we have tried feature engineering as an alternative to transfer learning approach.

BRDA used for feature optimization, has a number of parameters as shown in Table 1 which need to be specified at the start of the algorithm. The optimum values of these parameter has been determined experimentally following a trial and error approach. Each of these parameters have been plotted with respect to the corresponding accuracies obtained by the RF and MLP classifiers as shown in Figure 6. From the independent plots, we are able to determine the optimum values of these parameters. While determining a particular parameter, the remaining parameters are set to arbitrary values. This implies that each of these parameters, if set to their optimum values, can alone provide an optimized subset which can lead to accurate classification results which are close to the actual classification results when all the parameters with their optimum values are combined together. Therefore, a single parameter itself can bring a significant change to the entire model performance and parameter tuning becomes an

important step of experimentation.

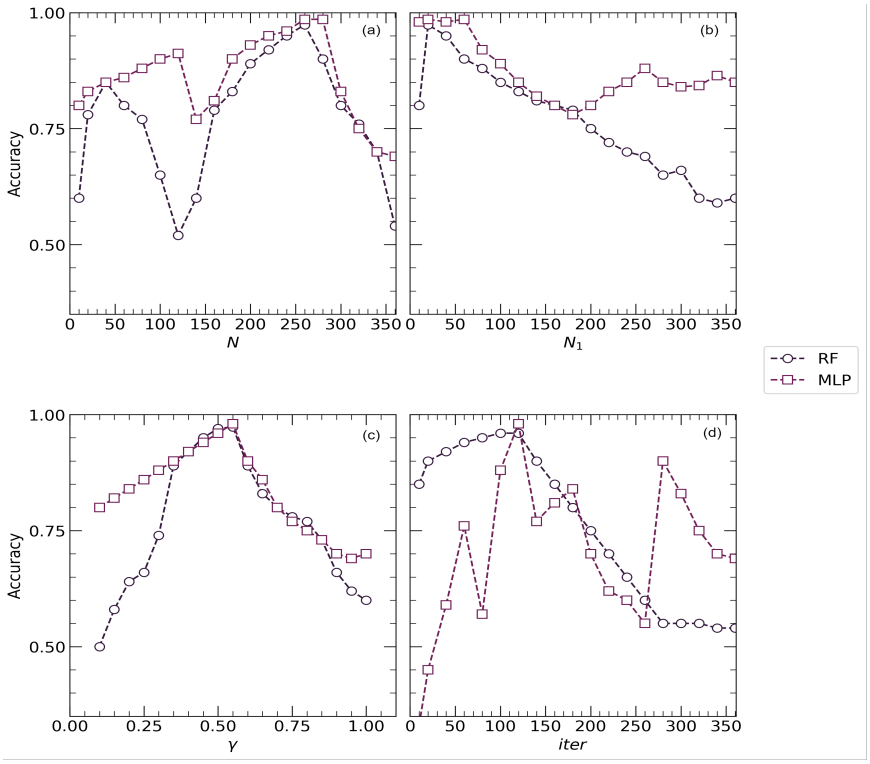

Figure 6: Experimental determination of the parameters of BRDA (a) RD population (N) (b) Number of male RDs (N_1) (c) Ratio of the number of commanders to total number of male RDs (γ) and (d) Total number of iterations for which BRDA is executed (iter)

Without BRDA, RF and MLP give poor classification results with accuracies of 0.4198 and 0.5556, respectively. As shown in Figure 7, the area under the ROC curves (AUC) for each of these classifiers without using BRDA are 0.478 and 0.532 respectively. These poor values of classification metrics are indication of improper classification of dendritic and non dendritic micrographs which can be seen from the true positive rate (TPR) and false positive rate (FPR) values as shown in Figure 8. TPR values are 0.4987 and 0.5073 and those of FPR are 0.775 and 0.8743 for RF and MLP classifiers respectively without using BRDA are However, after deploying the BRDA feature optimization algorithm, we note a drastic improvement in the classification results with accuracies of 0.9735 and 0.9855 and AUC of 0.97 and 0.979 for RF and MLP classifiers respectively.

Besides, the F1 scores for RF and MLP classifiers are 0.963 and 0.969 respectively after the implementation of BRDA in contrast to the same without BRDA

Table 1: Optimal parameter settings of the proposed FS method called BRDA

Parameter	Meaning	Value
N	RD population	260
N_1	Number of male RDs	20
γ	Ratio of the number of commanders to the total number of male RDs	0.57
$iter$	Total number of iterations for which BRDA is executed	120

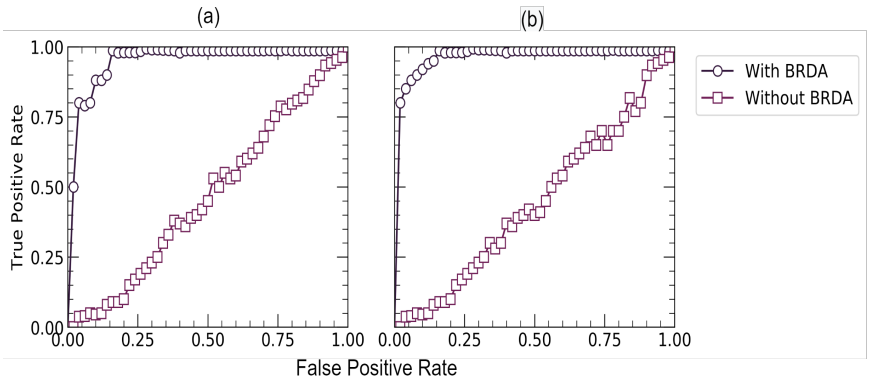

Figure 7: Performance of (a) RF and (b) MLP classifiers with and without using BRDA

which are 0.488 and 0.598 respectively. The F1 scores indicate that applying BRDA reduces the number of images falsely classified as the other category significantly. Figure 8 gives a comparative plot depicting the classifier evaluation metrics before and post implementation of BRDA. It has been established from the FPR values that the number of microstructural images which were incorrectly classified without using using BRDA has reduced significantly. FPR post implementation of BRDA are 0.39 and 0.25 for RF and MLP respectively. Figure 7 provides a better understanding on the performance of the classifiers with and without BRDA by the help of ROC curves.

4 Conclusion

This work focuses on a comparative study between transfer learning and feature engineering approaches in classification of dendritic and non-dendritic microstructural morphologies. Transfer learning using `VGG16`, `InceptionV3` and `Xception` fails to provide proper generalization in microstructural characterization. Consequently, we have utilized a feature engineering approach using a meta-heuristic optimization algorithm called Binary Red Deer Algorithm which

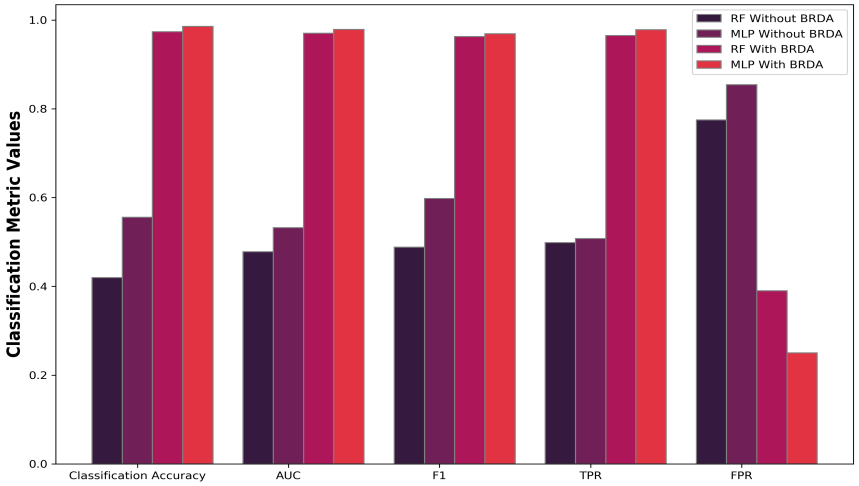

Figure 8: Comparison between the accuracies of RF and MLP classifiers with and without using BRDA

improves the performances of two machine learning classifiers drastically. Parameter tuning plays an important role in the optimization of the feature set and therefore impacts the classification results in turn. This work provides an uncommon example of where feature engineering has outperformed transfer learning. BRDA therefore brings significant promise in the field of materials informatics and we will aim to extend its use beyond materials characterization in future contributions.

5 Acknowledgements

The authors gratefully acknowledge support from the NSF CAREER Award DMR 1651668. Special thanks to Sterling Baird for his helpful comments and suggestions. The authors thank the creators of DoITPoMS for the creation of the microstructure databases used in this work. In addition, the authors express their gratitude to the open-source software community, for developing the excellent tools used in this research, including but not limited to Python, Pandas, NumPy, matplotlib, scikit-learn, and PyTorch.

References

[1] DTJ Hurle. “Constitutional supercooling during crystal growth from stirred melts—I: Theoretical”. In: *Solid-State Electronics* 3.1 (1961), pp. 37–44.

[2] W Wang, Peter D Lee, and M_ Mclean. "A model of solidification microstructures in nickel-based superalloys: predicting primary dendrite spacing selection". In: *Acta materialia* 51.10 (2003), pp. 2971–2987.

[3] Nikolas Provatas, Nigel Goldenfeld, and Jonathan Dantzig. "Efficient computation of dendritic microstructures using adaptive mesh refinement". In: *Physical Review Letters* 80.15 (1998), p. 3308.

[4] J Alkemper and PW Voorhees. "Three-dimensional characterization of dendritic microstructures". In: *Acta materialia* 49.5 (2001), pp. 897–902.

[5] Michael D Uchic. "Serial sectioning methods for generating 3D characterization data of grain-and precipitate-scale microstructures". In: *Computational methods for microstructure-property relationships*. Springer, 2011, pp. 31–52.

[6] Kaiqi Yang et al. "Self-supervised learning and prediction of microstructure evolution with convolutional recurrent neural networks". In: *Patterns* 2.5 (2021), p. 100243.

[7] Anh Tran et al. "An active learning high-throughput microstructure calibration framework for solving inverse structure–process problems in materials informatics". In: *Acta Materialia* 194 (2020), pp. 80–92.

[8] Xiaolin Li et al. "A transfer learning approach for microstructure reconstruction and structure-property predictions". In: *Scientific reports* 8.1 (2018), pp. 1–13.

[9] Xue Jiang et al. "An materials informatics approach to Ni-based single crystal superalloys lattice misfit prediction". In: *Computational Materials Science* 143 (2018), pp. 295–300.

[10] Arun Baskaran et al. "Adaptive characterization of microstructure dataset using a two stage machine learning approach". In: *Computational Materials Science* 177 (2020), p. 109593.

[11] Mohsen Ostad Shabani et al. "Refined microstructure of compo cast nanocomposites: the performance of combined neuro-computing, fuzzy logic and particle swarm techniques". In: *Neural Computing and Applications* 26.4 (2015), pp. 899–909.

[12] Aritra Chowdhury et al. "Image driven machine learning methods for microstructure recognition". In: *Computational Materials Science* 123 (Oct. 2016), pp. 176–187. DOI: 10.1016/j.commatsci.2016.05.034. URL: https://doi.org/10.1016/j.commatsci.2016.05.034.

[13] G. Impoco and L. Tuminello. "Incremental learning to segment micrographs". In: *Computer Vision and Image Understanding* 140 (Nov. 2015), pp. 144–152. DOI: 10.1016/j.cviu.2015.03.007. URL: https://doi.org/10.1016/j.cviu.2015.03.007.

[14] Filip Nikolić, Ivan Štajduhar, and Marko Čanaija. "Casting Microstructure Inspection Using Computer Vision: Dendrite Spacing in Aluminum Alloys". In: *Metals* 11.5 (2021), p. 756.

[15] Amir Mohammad Fathollahi-Fard, Mostafa Hajiaghaei-Keshteli, and Reza Tavakkoli-Moghaddam. "Red deer algorithm (RDA): a new nature-inspired meta-heuristic". In: *Soft Computing* 24.19 (2020), pp. 14637–14665.

[16] ZH Barber, JA Leake, and TW Clyne. "The DoITPoMS Project-a web-based initiative for teaching and learning materials science". In: *Journal of Materials Education* 29.1/2 (2007), p. 7.

[17] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[18] Christian Szegedy et al. "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[19] François Chollet. "Xception: Deep learning with depthwise separable convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1251–1258.

[20] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[21] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[22] Matt W Gardner and SR Dorling. "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences". In: *Atmospheric environment* 32.14-15 (1998), pp. 2627–2636.

[23] Masoud Fazli, Amir Mohammad Fathollahi-Fard, and Guangdong Tian. "Addressing a coordinated quay crane scheduling and assignment problem by red deer algorithm". In: *International Journal of Engineering* 32.8 (2019), pp. 1186–1191.

[24] Lutz Prechelt. "Early stopping-but when?" In: *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.

[25] Zijun Zhang. "Improved adam optimizer for deep neural networks". In: *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. IEEE. 2018, pp. 1–2.