IMPLEMENTING MACHINE LEARNING ALGORITHMS FOR IDENTIFYING MICROSTRUCTURE OF MATERIALS

A THESIS

Presented to the Department of Computer Engineering and Computer Science

California State University, Long Beach

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

Committee Members:

Thomas Johnson, Ph.D. (Chair)
Surajit Roy, Ph.D.
Todd Ebert, Ph.D.

College Designee:

Antonella Sciortino, Ph.D.

By Niyati S. Shah

B.Tech., 2013, Sardar Patel Institute of Technology, India.

August 2018

ABSTRACT

IMPLEMENTING MACHINE LEARNING ALGORITHMS FOR IDENTIFYING MICROSTRUCTURE OF MATERIALS

By

Niyati S. Shah

August 2018

Alloys of different materials are extensively used in many fields of our day-to-day life. Several studies are performed at a microscopic level to analyze the properties of such alloys. Manually evaluating these microscopic structures (microstructures) can be time-consuming. This thesis attempts to build different models that can automate the identification of an alloy from its microstructure. All the models were developed, with various supervised and unsupervised machine learning algorithms, and results of all the models were compared. The best accuracy of 92.01 ± 0.54% and 94.31 ± 0.59% was achieved, for identifying the type of an alloy from its microstructure (Task 1) and classifying the microstructure as belonging to either Ferrous, Non-Ferrous or Others class (Task 2), respectively. The model, which gave the best accuracy, was then used to build an Image Search Engine (ISE) that can predict the type of an alloy from its microstructure, search the microstructures by different keywords and search for visually similar microstructures.

ACKNOWLEDGEMENTS

Firstly, I would like to thank Dr. Surajit Roy, of the Mechanical and Aerospace Engineering Department at California State University, Long Beach, for giving me an opportunity to work on this project. Dr. Roy has always been helpful in identifying where I was going wrong and he made sure that all the basic concepts were made clear before implementing anything and helped me with any technical or functional issues whenever needed.

I would also like to thank Dr. Todd Ebert, who taught me Advanced Artificial Intelligence course during Spring 2017 semester, which sparked my interest in learning more about machine learning.

I want to thank my family for their unconditional support, unfathomable belief in me and for always pushing me to go one step further in everything I do.

Niyati Shailesh Shah

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ISE	Image Search Engine
DoITPoMS	Dissemination of Information Technology for the Promotion of Material Science
ASM	American Society for Metals
SIFT	Scale-Invariant Feature Transform
SVM	Support Vector Machine
HOG	Histogram of Oriented Gradients
Al	Aluminum
Cu	Copper
PCA	Principal Component Analysis
RF	Random Forest
RGB	Red Green Blue
VGG	Visual Geometry Group
ReLU	Rectified Linear Unit
SGD	Stochastic Gradient Descent
RMSProp	Root Mean Square Prop
AdaGrad	Adaptive Gradient
OOB	Out Of Bag

CHAPTER 1

INTRODUCTION

Materials characterization is a critical aspect of material design and discovery process [1]. We use many materials like aluminum, iron, steel, copper, and many more on a regular basis. However, apart from this daily use, these materials have a very crucial role to play in many engineering fields like construction, mechanical, aerospace, and so on. Each of these domains may demand different materials with different properties. Most of these materials are not pure metals, but they are alloys of metals and alloys are preferred because they are created in a way to make the metals stronger, harder, or lighter based on the need of a domain.

Microstructures of alloys are, images of the alloys, observed at a microscopic level. Studying various structures and patterns of these microstructures are very important because they depict very minute details about materials, like its tensile strength, elongation, and toughness. A case study, about the failure of liberty ships, shows the importance of studying the microstructures [2]. Around 2,708 liberty ships were constructed, from 1939 to 1945, and 1,301 damages or accidents were reported by 1946 [3]. This failure occurred because the material of the ship, when exposed to the frigid waters of the Atlantic, resulted in a formation of brittle cracks due to the lack of fracture toughness of the welded joint [2,3]. Thus, if a specific component is to be built of a particular material, predicting the behavior of that component under different conditions will require study of its microstructure under various phases.

Given a microstructure, manually studying its structure, understanding, and analyzing the pattern formation in those structures, and then predicting the type of an alloy requires a lot of expertise and man-hours. Some approach is thus needed, which can automate these steps. This thesis aims to develop different models, using machine learning techniques, which can

automatically identify the type of an alloy from its microstructure (Task 1) and classify them as either belonging to Ferrous, Non-Ferrous or Others class (Task 2).

Machine learning is a part of artificial intelligence which focuses on the development of models that can be trained on some data and then use the trained model in future to predict the new data. There are many different types of models in machine learning like classification model, regression model, neural network model and many more. This thesis aims to build the neural network models for both Task 1 and Task 2. Neural networks are biologically inspired programming method, which learns to process the data as humans do. Neural networks do not need any prior knowledge about the microstructures as they can learn features, from the input provided, during their training process.

Objective

Based on the above-introduced concepts, the goals of this thesis are as follows:

- To gather microstructures of alloys, from various sources, which are obtained under different phases and at different magnification factor.
- To preprocess these microstructure images for using them as an input to the neural network models.
- To build and train different neural network models for Task 1 and Task 2.
- To compare and evaluate the performance of all the models.
- To build an Image Search Engine (ISE).

Organization of Thesis

The organization of thesis is as follows. Chapter 2 does the literature survey of some research papers and provides the details of additional work done in this thesis. Chapter 3 provides an overview of how the dataset was created, from raw microstructural images, and how

image augmentation technique was applied to increase the size of the dataset. Chapter 4 provides a detailed explanation of the machine learning techniques, used in this thesis, for building neural network models. Chapter 4 also provides details about how different parameters were selected, which can be used to train the models. Chapter 5 gives the detailed description of each model and how much accuracy was achieved, by each model, for Task 1 and Task 2. This chapter also compares results of all the models and highlights the model with the best accuracy. Chapter 6 shows the performance of various functions of ISE and gives a detailed explanation of how visually similar images were selected. Chapter 7 provides the software and hardware specifications, used in this thesis, for the training of models and for building the ISE. Chapter 8 gives the conclusion for the entire thesis work and highlights on the future work.

CHAPTER 2

LITERATURE REVIEW

The following research papers provide a brief explanation of the machine learning techniques, which were previously implemented, for studying the microstructures of materials.

Image Driven Machine Learning Methods for Microstructure Recognition [1]

This research paper performed a case study on the dendritic morphology of the microstructures. They classified the microstructures as having dendrites or not, and those having dendrites were later classified as having either longitudinal or transverse cross-section views.

Dataset, for this research paper, included microstructure images (micrographs) with a different magnification factor, a different composition, and a different orientation of the microstructural features. It included 528 micrographs obtained from the Dissemination of Information Technology for the Promotion of Material Science (DoITPoMS) library.

In this paper, authors had used different combinations of feature extraction, feature selection, and classification techniques, for the training of models, and classification was completed using full images and reduced feature vectors. Comparing the performances of all the models, the best accuracy achieved, in this paper, was 91.85 ± 4.25% for the classification of the microstructures depicting or not the dendritic morphology.

In this thesis work, rather than just identifying if the microstructure has a dendrite or not, the trained neural network model was able to identify the type of an alloy from a given microstructure and then classify it into a class of Ferrous, Non-ferrous or Others. The dataset, for this thesis, included images from both DoITPoMS library and American Society for Metals (ASM) library and had 4,992 micrographs in total, thus having more data for training. For building the models, apart from feature extraction and classification layer, this thesis also

included extra hidden layers. Comparing the performances of all models, the best accuracy achieved, in this thesis, was 92.01 ± 0.54 % and 94.31 ± 0.59% for Task 1 and Task 2 respectively.

A Computer Vision Approach for Automated Analysis and Classification of Microstructural Image Data [4]

The goal of this paper was to build an ISE which provided the best matches for a query image in a database of microstructures. The Visual Bag of Words (VBoW) was used, along with the Scale-Invariant Feature Transform (SIFT) as a feature extraction technique, and these features were then used to find visually similar micrographs.

The dataset, in this research paper, included only 105 microstructural images manually collected from internet. These micrographs were classified, into seven distinct classes: brass, ductile cast iron, grey cast iron, hypo eutectoid steel, malleable cast iron, superalloy, and annealing twins.

The model, in this paper, was trained using a Support Vector Machine (SVM) classifier with a 5-fold cross-validation technique. The accuracy achieved for this classification model was 83.00 ± 3.00%. The output of a visual microstructure search engine was the top four matches for the query image.

In this thesis, instead of the SIFT descriptor, Histogram of Oriented Gradients (HOG) was used to identify critical points in micrographs and then use this feature vector to search for visually similar micrographs. HOG was selected because it is a dense feature extractor wherein it extracts features from all the locations in an image whereas SIFT descriptor extract features only from the neighborhood of critical points (region of interest). An ISE was developed, in this thesis, which can fulfill multiple purposes as follows: predict the type of an alloy from its

microstructure, search microstructures by keywords, and search for visually similar microstructures.

The dataset for building ISE, in this thesis, included the micrographs obtained from both DoITPoMS and ASM library. Next chapter gives a detailed explanation about how the dataset, used in this thesis, was created.

CHAPTER 3

DATASET

The micrographs, used in this thesis, were obtained from two sources, ASM International Micrograph Database and DoITPoMS library. ASM International is a material science library which has many different datasets like ASM alloy phase diagram database, ASM medical materials database, and ASM micrograph database. ASM micrograph database was used, in this thesis, which has approximately 4,600 micrographs of different alloy materials. DoITPoMS is the publicly available library which has nearly 500 micrographs.

The micrographs from both the libraries were combined to create the dataset for this thesis. The micrographs in DoITPoMS library already present in ASM library were discarded. So, the total number of micrographs combined from both, DoITPoMS and ASM, libraries were 4,992 micrographs.

The ASM library already had micrographs divided into below 19 classes as follows: general, ceramic, composite, dissimilar metal weldment, iron, steel, aluminum, cobalt, copper, lead, magnesium, miscellaneous, nickel, refractory, superalloy, tin, titanium, zinc, and polymer.

In DoITPoMS library the micrographs were not directly classified like in ASM dataset. In this thesis, to place those micrographs in ASM classes, each micrograph in DoITPoMS library was checked for its composition. So, based on its composition, the micrographs in DoITPoMS library were merged with the categories of ASM dataset. For example, the micrograph with a composition of Aluminum (Al): 75% and Copper (Cu): 25%, was included in the category of aluminum and the micrograph with a composition of Al:10% and Cu: 90%, was included in the category of copper. The classes, of general and dissimilar metal weldment, were discarded because they had only 2-3 micrographs which are decidedly less for training a model. In this

thesis, a category of the polymer was merged, with a category of the composite. Thus, the dataset (Dataset 1), for this thesis, included 4,992 micrographs with 16 different categories collected from both ASM and DoITPoMS library.

All the micrographs, in both the libraries, had a scale bar and some of them had their compositions written on it. Thus, the micrographs, used in this thesis, were cropped to remove the scale bar and edited to remove the compositions written on it.

Most of the machine learning algorithms give good results if the dataset has uniformly distributed data for all the classes. From the total of 4,992 micrographs, 2,003 micrographs were of steel, 500 micrographs were of iron, 549 micrographs were of aluminum, 588 micrographs were of copper, and the rest 1,352 micrographs were divided unequally amongst the remaining 12 classes. In this thesis, to ensure uniform distribution, the micrographs in all the categories except for steel were augmented (flips and rotation at different angles). The size of the dataset was thus increased from 4,992 micrographs to 13,475 micrographs, each class having approximately 1,000 – 1,200 micrographs. The micrographs of cobalt, lead, nickel, tin, and zinc were discarded, in this thesis, because the original dataset had only 35, 32, 40, 12 and 31 micrographs respectively for each class and even after image augmentation, these classes would have approximately 200 – 300 micrographs which would be decidedly less as compared to the other categories. Thus, the new dataset (Dataset 2) had a count of 13,475 micrographs with 11 distinct categories of alloy material.

In this thesis, the 16 categories from Dataset 1 and 11 categories from Dataset 2, were also further grouped into three categories as required for Task 2. The category of Ferrous included the micrographs of iron and steel, the category of Non-Ferrous included the micrographs of aluminum, cobalt, copper, lead, magnesium, misc., nickel, refractory, superalloy,

tin, titanium, and zinc, and the category of Others included the micrographs of ceramic and composite.

All the models, in this thesis, were built and trained using either Dataset 1 or Dataset 2 and a detailed explanation of all the machine learning techniques, used for building and training these models, are presented in the next chapter.

CHAPTER 4

MACHINE LEARNING TECHNIQUES

All the models, developed in this thesis, for both Task 1 and Task 2 follow the same underlying architecture as shown in Figure 1.

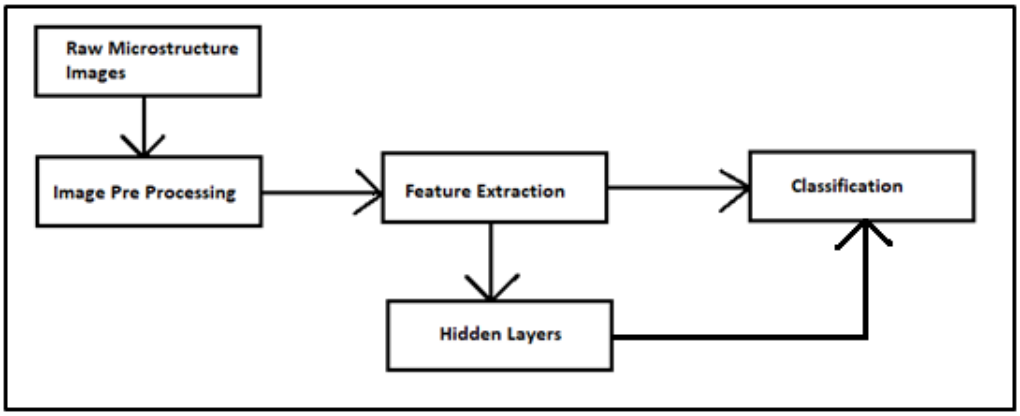

FIGURE 1. **Basic architecture of all models.**

Figure 1 shows that the model takes a raw micrograph as an input, which is then preprocessed and essential features are extracted, at the feature extraction layer. These reduced feature vectors are then connected, to a hidden layer and a classification layer. Some models may or may not have a hidden layer, so the extracted features are directly classified.

In this thesis, different feature extraction methods like pre-trained neural networks, autoencoders, and Principal Component Analysis (PCA), were used. Different type of hidden layers like fully connected layer, batch normalization layer, convolution, and max-pooling layers, were used. The classifiers used were Softmax and Random Forest (RF), and for the training of the models, K-fold cross-validation technique was used.

In the task of image recognition, model learns the information from images pixel by pixel. All the micrographs, used in this thesis, were colored images having pixel values for all the three: Red, Green, and Blue (RGB) channels. For example, if a colored image has a resolution of

28 by 28, the total number of pixels/dimensions/features in that image would be 2,352 (28 x 28 x 3). Most of the images, in both Dataset 1 and Dataset 2, were of resolution 1000 by 500, so the total number of input pixels were 1,500,000 (1000 x 500 x 3). Providing such a substantial input for training a model, which includes going back and forth to use the input values, would be computationally expensive. Feature extraction methods were thus used to reduce this complexity.

Feature extraction / Feature selection is a method of reducing the dimensions from the input data without losing relevant information from the data. Most of the information in raw pixel data is redundant, and thus it can be reduced to only essential features. Determining a subset of primary features is called feature selection [5]. Selecting a good feature extraction algorithm is very important as it directly affects the performance of a model.

Both supervised and unsupervised machine learning techniques were used, in this thesis, for feature extraction and dimensionality reduction purposes.

Supervised Learning Method for Feature Extraction

Supervised machine learning is the construction of algorithms that can produce general patterns and hypotheses by using externally supplied instances to predict the fate of future instances [6]. In simpler terms, supervised learning is a method where the model learns based on the input provided and the labels assigned to these inputs. In this thesis, the labels were 11/16 classes for Task 1 and three classes for Task 2. Equation (1) shows the basic formula for supervised learning method.

$$Y = f(X) \tag{1}$$

Given an input X (image pixels) and a label Y (class) assigned to it, the model learns the mapping function given by Equation (1). The learned mapping function is then used to predict the class of a new data.

One of the feature extraction methods, used in this thesis, was using the features learned from the pre-trained neural network. This pre-trained neural network was trained, in a supervised learning way, where it had a labeled training data. The type of pre-trained neural network used, in this thesis, was a ResNet50 model. ResNet50 model was trained, on an ImageNet dataset which approximately has 1.2 million training images, 50,000 validation images, and 150,000 testing images of many classes (labels) like cats, dogs, cars, ships, airplanes, and many more natural images [7]. The ResNet50 architecture achieved as low as 3.57% error on the ImageNet test set [8].

The ResNet50 architecture had 50 hidden layers including convolutional layer, max pooling layer, and fully connected layer. Table 1 shows the architecture of the ResNet model.

TABLE 1. ResNet Architectures [8]

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

The convolution layers extract the features from image pixels, in a such a way that they preserve the spatial relationship between pixels, by learning the features grid by grid and pooling layers, on the other hand, are used for downsampling purpose. In fully connected layers all the pixels in preceding layers are connected to all the pixels in the next layer. From Table 1 it is seen

that conv2_x, conv3_x, conv4_x, and conv5_x has a stacked block of convolution layer. Figure 2 shows the representation of such block function.

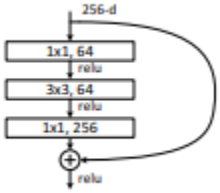

FIGURE 2. Residual blocks representation of ResNet architectures [8].

Python keras library was utilized to use the pre-trained ResNet50 architecture, as a feature extraction method, in this thesis. Keras is an open source neural network library, which has pre-trained weights of many architectures like Visual Geometry Group (VGG) 16, VGG19, and ResNet50. The ResNet50 model takes a colored image of resolution 224 by 224 as an input, and thus the micrographs, from both Dataset 1 and Dataset 2, were reshaped. In this thesis, the features were extracted from a conv5_x layer of the ResNet50 model, so the output from this layer was a feature vector of size 2048, as shown in Table 1.

Un-Supervised Learning Method for Feature Extraction

As compared to supervised learning methods, the unsupervised learning methods have unlabeled training data. Unsupervised learning algorithms learn by discovering different structures and pattern in the data. The unsupervised learning algorithms used, in this thesis, were autoencoders and PCA.

The conventional autoencoder is a three-layer symmetrical neural network that constrains the output to be equal to the input [9]. Figure 3 represents the basic layout of an autoencoder.

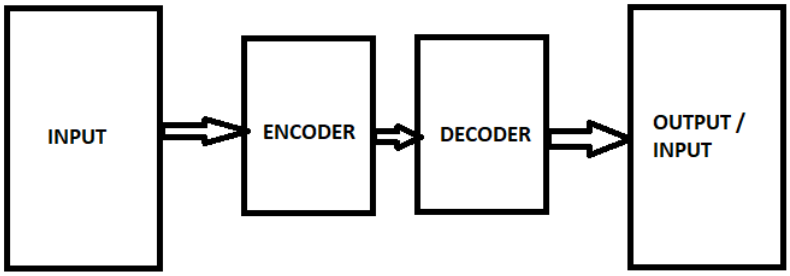

FIGURE 3. A Basic layout of an autoencoder.

The learned encoded and decoded data, of an autoencoder, can be used as a feature vector. The encoder part can be build using a simple single layer, or it can be created using multiple layers. In this thesis, encoder was designed using three convolution layers and two max-pooling layers. The decoder is the reverse of layers used in an encoder. Table 2 gives a detailed explanation of both the encoder and decoder layers used, in this thesis, for building an autoencoder.

TABLE 2. Representation of Hidden Layers in Autoencoder

Layer Name	Input size	Output Size	Layer description
Encoder			
conv_1	128 x 128 x 1	128 x 128 x 8	8, (5 x 5) filters, with stride 1
maxpool_1	128 x 128 x 8	64 x 64 x 8	2 x 2 maxpool
conv_2	64 x 64 x 8	64 x 64 x 16	16, (5 x 5) filters, with stride 1
maxpool_2	64 x 64 x 16	32 x 32 x 16	2 x 2 maxpool
conv_3	32 x 32 x 16	8 x 8 x 32	32, (5 x 5) filters, with stride 4
Decoder			
conv_3T	8 x 8 x 32	32 x 32 x 32	32, (5 x 5) filters, with stride 4
upsample_2	32 x 32 x 32	64 x 64 x 32	2 x 2 up sampling
conv_2T	64 x 64 x 32	64 x 64 x 16	16, (5 x 5) filters, with stride 1
upsample_1	64 x 64 x 16	128 x 128 x 16	2 x 2 up sampling
conv_1T	128 x 128 x 16	128 x 128 x 1	1, (5 x 5) filters, with stride 1

For the training of an autoencoder, grayscale micrographs from, Dataset 2 was used. The micrographs were converted to grayscale to learn the features in it and not the colors. The output of an encoded function was the total number of pixels from the conv_3 layer, as shown in Table

2, which is 2048 (8 x 8 x 32). Using the decoded output, as a feature vector, the size of the vector is same as the number of dimensions in an input image which is 16,384 (128 x 128 x 1), as shown in Table 2.

PCA is a dimensionality reduction technique that finds the principal components of data in the direction where the input data is more spread out. PCA does not select specific features and discards others, but it creates a new set of features summarizing the details of other features in total. PCA computes variance for all the features in an image. In this thesis, the number of features which gave 99.9% variance was selected. The input for PCA algorithm was 100 by 100 colored micrographs, and thus, at the preprocessing layer, the micrographs were reshaped to this required resolution. The total number of input features for PCA, in this thesis, was 30,000 (100 x 100 x 3), and PCA gave a variance of 99.9% for 768 features. Thus, the dimension of the input data, in this thesis, was reduced to from 30,000 features to 768 features.

The reduced feature vector, from any of the feature extraction method, was provided as an input to hidden layers or directly to classification layer. Different classifiers were used, in this thesis, for classification layer.

Classifiers

After feature extraction, the extracted features were connected to the hidden layers and then classified. Classifier, as the name suggests, is for classifying the input as belonging to a particular class. In this thesis, there were 11/16 classes for Task 1 and three classes for Task 2. There are many classifiers in machine learning, like Naive Bayes classifier, SVM, Decision Trees, RF, Softmax and many more. The classifiers used, in this thesis, were Softmax and RF.

Softmax classifier outputs the probabilities of each class, for a given input. For example, for Task 1, the Softmax classifier will output different probabilities for all the 11/16 classes. The

input image is predicted to be in a class with the highest probability. Equation (2) represents the formula for a multi-class Softmax classifier.

$$H_\theta(x) = \begin{bmatrix} P\ (label = 1\ |x; \theta) \\ P(label = 2\ |x; \theta) \\ \vdots \\ P(label = K\ |x; \theta) \end{bmatrix} \quad (2)$$

In Equation (2), x is an input given to the Softmax classifier, K is the total number of classes and θ is the weight. The output is a K-dimensional vector, giving K estimated probabilities. The term, P (label =1|x; θ) denotes the probability of an input belonging to class 1, parameterized by θ. All the probabilities are computed, and the final predicted class is the one with the highest probability value.

RF algorithm follows the divide and conquer approach. RF creates multiple decision trees which, at each level, questions about the features and outputs the answer whether the feature is present or not. Organization of the decision trees is in a hierarchical manner, where the root node is the start of the decision tree which represent a single feature, internal nodes also represent features, which may or may not be, connected to the root nodes, and each leaf node represents the class label. This structure of decision tree predicts the class label as an output. Multiple decision trees will output multiple class label for a single image, but the final class is that which is predicted by most of the decision trees.

In this thesis, the number of estimators or decision trees were selected to be 1,000, the minimum number of samples required to split the internal nodes was 43, and the minimum number of samples required at the leaf node was 45.

After selecting the feature extraction technique, hidden layers, and a classifier, a model needs to be trained.

For the training of a model, the dataset was divided, into a training set, a validation set, and a test set. The training set was an input given to a model. Evaluation of the performance, of a trained model, was done on the validation set, and if the validation accuracy was low, the training parameters were re-tuned, and the model was re-trained. The final accuracy was the evaluation of a model on the test set. Various parameters were configured, in this thesis, for the training of a model to achieve reasonable accuracy for both Task 1 and Task 2.

The training parameters, selected in this thesis, includes activation functions (Rectified Linear Unit (ReLU)), Error functions (cross entropy), optimizers (Adam (Adaptive Moment Estimation)), and training methods (K-Fold Cross Validation).

Activation Functions

An activation function follows most of the hidden layers in the neural network models. The primary purpose of an activation function is to convert the input from the previous layer into some non-linear output, which is an input to the next layer. There are many different activation functions like Sigmoid, Tanh, ReLU, Leaky ReLU, and so on. The activation function used, in this thesis, was ReLU because previous research showed that ReLU's results in much faster training [7].

ReLU activation function is the simplest non-linear activation function which is given by Equation (3).

$$f(x) = \max(0, x) \tag{3}$$

The value x is the input given to the ReLU function. The function computation is simple, either 0 (not activated) or x (activated), depending on the sign of x. Figure 4 shows the plot of ReLU function.

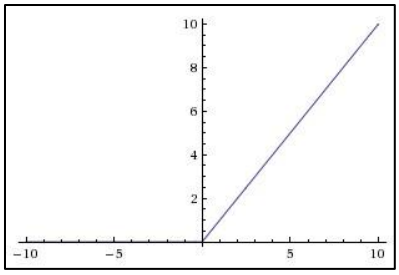

FIGURE 4. Graphical representation of ReLU function.

Error Functions

Error function measures the accuracy, or an error based on the difference between the original output and the output predicted by a model. There are many different loss functions, but they are selected based on the classifier used. In this thesis, for the models which use the Softmax classifier, error function used was cross entropy loss and for the models which use the RF classifier the error criterion used was an Out-of-bag error.

A cross entropy loss measures the performance of a model whose output is the probabilities for all the different classes. Equation (4) shows the multi-class cross entropy loss function.

$$-\sum_K y log(\hat{y}) \tag{4}$$

In Equation (4), y is the correct class label, and $\hat{y}$ is the predicted label. The output of this function gives an accuracy of how far the predicted value from the actual value is.

OOB is also a prediction error function. For each estimator run, the RF classifier randomly partitions the dataset into a train set and a validation set, internally. Thus, in this thesis, the models which used RF classifier did not need any external cross-validation method for its training. During each estimator run, one-third of the dataset was used as a validation set. At the end of each estimator run the class which receives majority votes is tested, on the validation set.

The proportion of times, the predicted class is not equal to correct class, is averaged and that averaged value is the OOB error estimate.

Optimizers

After finding the error, some optimization method was needed, which could be used to minimize the error function. Optimizers are used to update the weights in the neural network, during the training process, to minimize the error function. There are many optimizers like Stochastic Gradient Descent (SGD), Root Mean Square prop (RMSprop), Adam, Adadelta, Adaptive Gradient (AdaGrad), and so on. In this thesis, an Adam optimizer was selected, because Adam works well in practice and compares favorably to other stochastic optimization methods [10]. Equation (5), Equation (6), and Equation (7) represent the formula for an Adam optimizer.

$$m = \beta 1 * m + (1 - \beta 1) * dx \tag{5}$$

$$v = \beta 2 * v + (1 - \beta 2) * (dx ** 2) \tag{6}$$

$$x += \frac{-\alpha * m}{\epsilon + \sqrt{v}} \tag{7}$$

Some modifications were made, in this thesis, for an Adam optimizer. In Equation (7), the learning rate, α, was changed from the default value of 0.001 to 0.0001, to adapt slow learning. In Equation (5), the exponential decay rate for the first moment, $\beta 1$, used the default value of 0.9. In Equation (6), the exponential decay rate for the second moment, $\beta 2$, used the default value of 0.999. In Equation (7), a minimal number to prevent division by zero, ϵ, was changed from the default value of 10e-8 to 2e-11.

Training Methods

After selecting all the parameters value, a model must be trained. The models which used RF classifier, in this thesis, did not have any specific training method because RF classifiers

internally train themselves during each run of an estimator. Other models that used the Softmax classifier, in this thesis, were trained using k-fold cross-validation technique.

Cross-validation is a method in which the models are trained by partitioning the input dataset into a training set and a test set. The training set is then further divided into a train set and a validation set. The cross-validation technique used in this thesis was k-fold cross-validation, where k was either five or eight. Figure 5 shows the visualization of a 5-fold cross-validation technique.

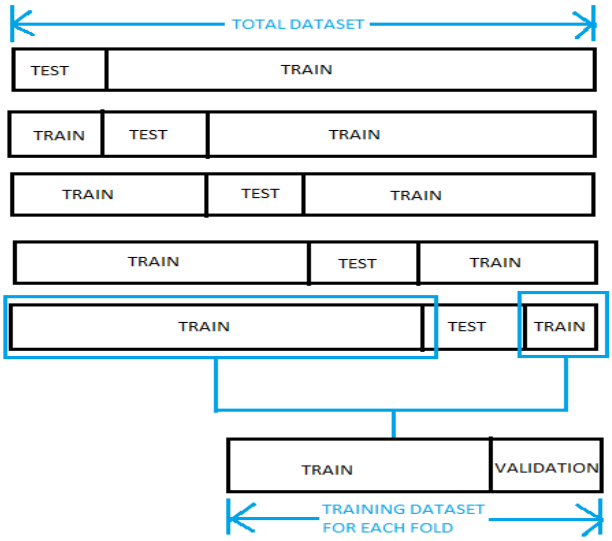

FIGURE 5. 5-fold cross validation representation.

During the training of a model, an error function is minimized, for the total number of iteration or epoch on each fold. In this thesis, the dataset was divided, into a 75% train set and a 25% test set. The train set for each fold was divided, into an 85% train set and a 15% validation set. The dataset was shuffled while dividing the dataset into a train set and a test set. The validation accuracy was computed, after each epoch, and the test accuracy was computed, after each fold. Average of the test accuracy across all the folds, was the final model accuracy.

CHAPTER 5

MODELS AND RESULTS

In this thesis, different models were built, using the machine learning techniques presented in Chapter 4, and results of all the models were compared. This chapter provides a brief overview of the different models built in this thesis, how much time did each model take for training and what was the accuracy achieved by each model for both Task 1 and Task 2.

Model 1 was built using the Dataset 2 containing 13,475 micrographs. The feature extraction method used for this model was PCA, and the number of input features was reduced from 30,000 to 768 with a variance of 99.9%. Table 3 shows the representation of the hidden layers used in Model 1. The classifier used for Model 1 was a Softmax classifier.

TABLE 3. Representation of Hidden Layers in Model 1

Layer Name	Input size	Output Size	Layer description	Activation
conv_1	16 x 16 x 3	16 x 16 x 96	96, (11 x 11) filters	Relu
maxpool_1	16 x 16 x 96	5 x 5 x 96	3 x 3 maxpool	NA
batchnorm_1				
conv_2	5 x 5 x 96	5 x 5 x 256	256, (5 x 5) filters	Relu
maxpool_2	5 x 5 x 256	1 x 1 x 256	3 x 3 maxpool	NA
batchnorm_2				
conv_3	1 x 1 x 256	1 x 1 x 384	384, (3 x 3) filters	Relu
conv_4	1 x 1 x 384	1 x 1 x 256	256, (3 x 3)	Relu
FC_1	256	512	fully connected	Relu
FC_2	512	128	fully connected	Relu
FC_3	128	11 (for task 1), 3 (for task 2)	fully connected	Softmax

In Table 3, the hidden layer of batch normalization is a layer which normalizes the input from the previous layer. Normalization means maintaining the activations of preceding layers with a mean close to 0 and the standard deviation close to 1. Model 1 was trained with an 8-fold cross-validation technique with 50 epochs and a batch size of 100. The time taken to train this

model was 58 seconds for each epoch, so the total training time was approximately 6 hours. The accuracy of Model 1 was 58.36% with a standard deviation of ± 1.31%, for Task 1 and the accuracy of 74.36 % with a standard deviation of ± 1.8%, for Task 2. Model 1 gave a reasonable accuracy for Task 2, but a very low accuracy for Task 1.

So, Model 2 and Model 3 were built using convolutional autoencoders as the feature extraction method. Model 2 and Model 3 both used Dataset 2 for Task 1 and Task 2.

Model 2 was built using 2,048 features extracted from the encoded layer, as shown in Table 2, of the trained autoencoder. There were no additional hidden layers for this model. The model was classified and trained using RF. The total training time for this model was approximately 1 hour. The accuracy achieved for this model was 50.35% for Task 1 and 60.53% for Task 2. Model 2 gave a very low accuracy for both Task 1 and Task 2 as compared to Model 1.

Model 3 was built using 16,384 features extracted from the decoded layer, as shown in Table 2, of the trained autoencoder. Rest all the configuration was same as for Model 2. The total training time for Model 3 was approximately 3 hours. The accuracy achieved for Model 3 was 52.96% for Task 1 and 61.27 % for Task 2. Model 3 gave a slightly better accuracy as compared to Model 2, but still low accuracy as compared to the Model 1.

All these models were built based on the unsupervised feature extraction method. Model 4 and Model 5 were built using the supervised feature extraction method.

Model 4 was built using the Dataset 2. The feature extraction method used for this model was to use the saved weights from the pre-trained ResNet50 model, which reduced the number of features from 152,528 (224 x 224 x 3) to 2,048. These reduced feature vectors were then

connected, to the multiple hidden layers. The representation of hidden layers for Model 4, is shown in Table 4. Model 4 was classified using Softmax classifier.

TABLE 4. Representation of Hidden Layers in Model 4

Layer Name	Input size	Output Size	Layer description	Activation
FC_1	2048	200	fully connected	Relu
FC_2	200	11 (for task 1), 3 (for task 2)	fully connected	Softmax

Model 4 was trained with an 8-fold cross-validation technique with 500 epochs and a batch size of 100. The time taken to train this model was 1 second for each epoch, so the total training time was approximately 1 hour. The accuracy of Model 4 for Task 1 was 92.01% with a standard deviation of ± 0.54%, and for Task 2 the accuracy was 94.31% with a standard deviation of ± 0.59%.

Model 4 gave the best accuracy for both Task 1 and Task 2, as compared to Model 1, Model 2, and Model 3. Since this model gave good accuracy for Dataset 2, a new model similar to Model 4, was built using Dataset 1.

Model 5 was built using the same feature extraction method as used in Model 4. The hidden layer representation for Model 5 is given by Table 5.

TABLE 5. Representation of Hidden Layers in Model 5

Layer Name	Input size	Output Size	Layer description	Activation
FC_1	2048	200	fully connected	Relu
FC_2	200	16 (for task 1), 3 (for task 2)	fully connected	Softmax

Model 5 was trained with a 5-fold cross-validation technique with 500 epochs and a batch size of 75. The time taken to train this model was 0.5 second for each epoch, so the total training time was approximately 20 minutes. The accuracy of Model 5 for Task 1 was 79.66% with a standard deviation of ± 1.65% and the accuracy for Task 2 was 70.73% with a standard deviation

of ± 1.17%. This model did not give a good accuracy as compared to Model 4, but it still gave better accuracy as compared to the accuracies of Model 1, Model 2, and Model 3.

Results

Table 6 shows the comparison of all the models in a tabular form and the model having the best accuracy, for both Task 1 and Task 2, is highlighted.

TABLE 6. Results of Different Models

Models	Total Micrographs	Total Classes	Original Number Of Features	Reduced Number Of Features	Validation Accuracy	Test Accuracy
Model1_Task1	13,475	11	30,000 (100 x 100 x 3)	768 (16 x 16 x 3)	(56.36%, 58.97%, 58.57%, 56.12%, 58.92%, 58.86%, 60.20%, 58.89%)	58.36%(+-1.31%
Model1_Task2	13,475	3	30,000 (100 x 100 x 3)	768 (16 x 16 x 3)	(76.22%, 73.53%, 76.80%, 74.7%, 75.3%, 73.1%,74.47%,70.71%)	74.36%(+-1.8%)
Model2_Task1	13,475	11	16,384 (128 x 128 x 1)	2048	0.5329	50.35%
Model2_Task2	13,475	3	16,384 (128 x 128 x 1)	2048	0.6519	60.53%
Model3_Task1	13,475	11	16,384 (128 x 128 x 1)	16,384 (128 x 128 x1)	0.5634	52.96%
Model3_Task3	13,475	3	16,384 (128 x 128 x 1)	16,384 (128 x 128 x1)	0.6299	61.27%
Model4_Task1	**13,475**	**11**	**150,528 (224 x 224 x 3)**	**2048**	**(92.3%, 92.6%, 92.23%, 91.09%, 91.8%, 91.5%, 91.79%, 92.8%)**	**92.01% (+-0.54%)**
Model4_Task2	**13,475**	**3**	**150,528 (224 x 224 x 3)**	**2048**	**(93.83%, 93.71%, 94.78%, 94.42%, 94.42%, 95.31%, 94.66%, 93.40%)**	**94.31% (+-0.59%)**
Model5_Task1	4,992	16	150,528 (224 x 224 x 3)	2048	(71.61%, 68.43%, 70.94%, 71.18%, 71.47%)	70.73% (+-1.17%)
Model5_Task2	4,992	3	150,528 (224 x 224 x 3)	2048	(80.56%, 81.16%, 76.95%, 79.32%, 78.92%)	79.66% (+-1.65%)

CHAPTER 6

IMAGE SEARCH ENGINE

The ISE was developed, in this thesis, like a python application which has many different functions. Figure 6 shows the main screen of ISE.

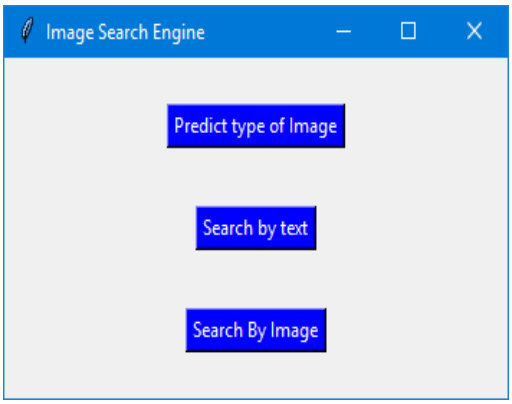

FIGURE 6. The main screen of Image Search Engine.

The first function of ISE was to predict the type of an alloy of a given micrograph. The prediction was made using Model 4 which gave the best accuracy for both Task 1 and Task 2. Figure 7 shows the predicted type of alloy and its prediction score, for both Task 1 and Task 2, for three different micrographs.

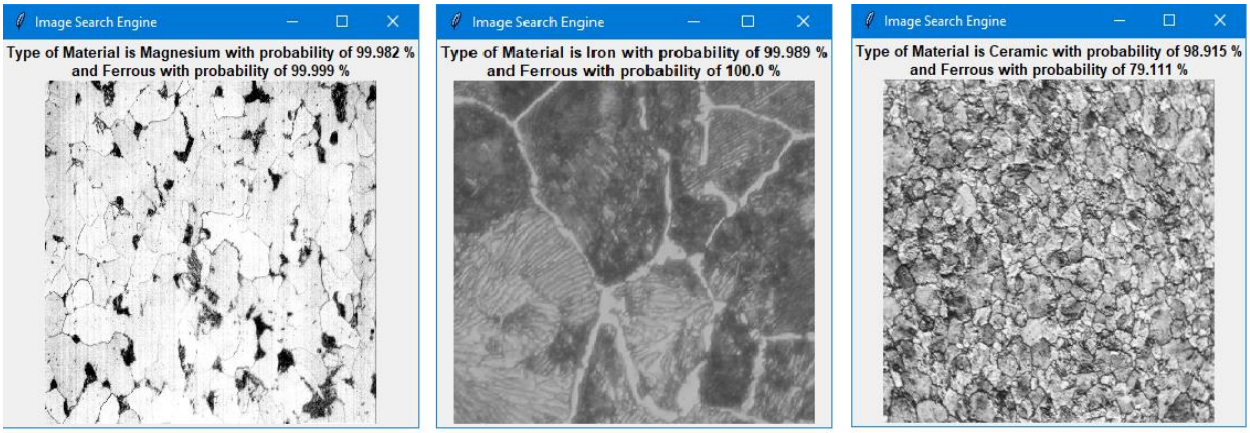

FIGURE 7. Prediction score of different micrographs.

The second function of ISE was to search by text, wherein top three micrographs were searched and displayed, based on its alloy type. Figure 8 shows the search screen.

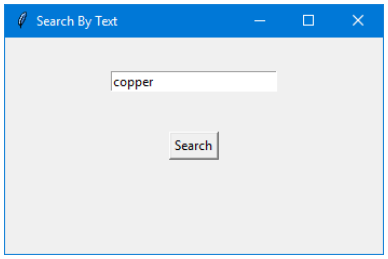

FIGURE 8. Search screen of Image Search Engine

Figure 9 shows the search results for copper, misc., and refractory, displaying three different micrographs for each of the classes.

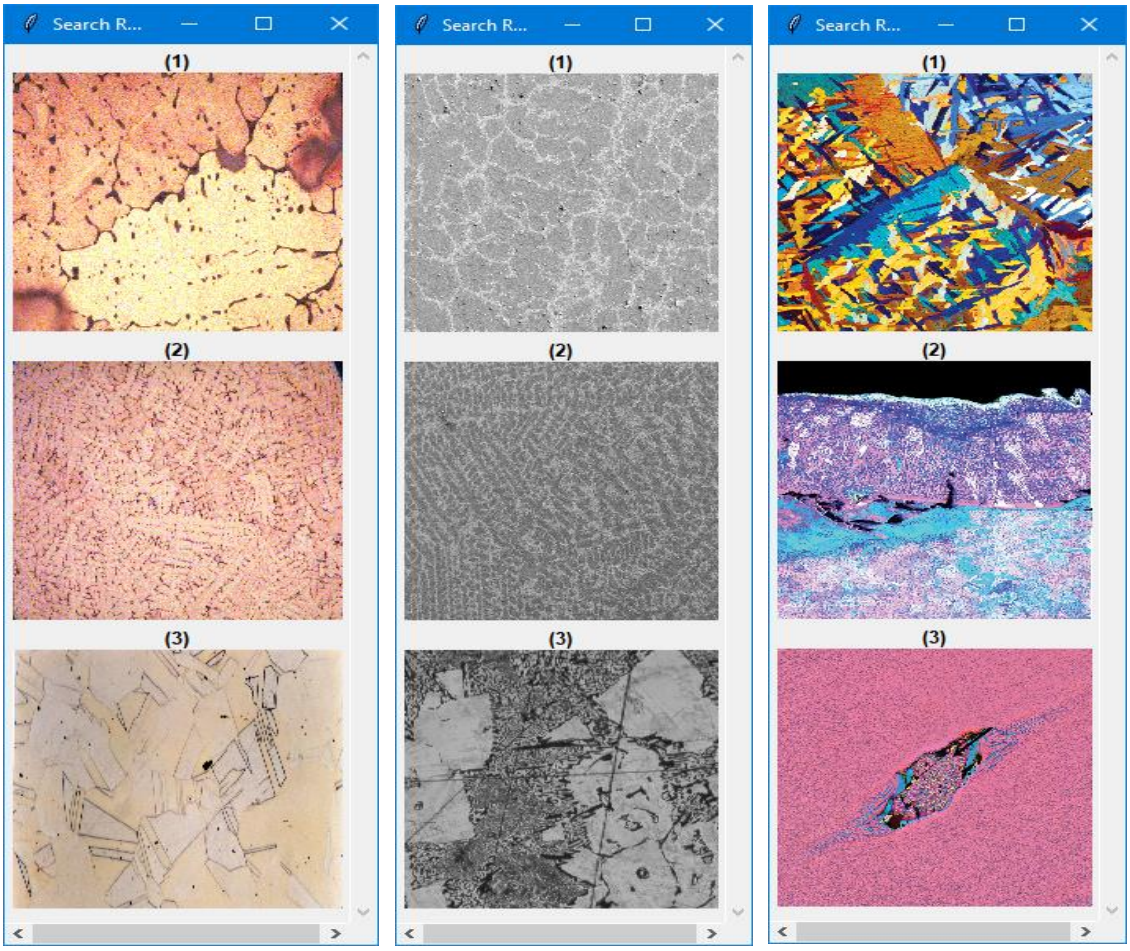

FIGURE 9. Micrograph search results for copper, misc., and refractory.

The final function of the ISE was to find top five visually similar micrographs, given a query micrograph and then predict the type of each micrograph. In this thesis, for prediction, Model 4 was used, and HOG was used to search for visually similar micrographs.

The HOG is often used, for object detection and feature extraction method. Given the pixels of an image, we can try to find how much intensity value changes in each direction. In HOG feature extraction technique, the distribution (histograms) of a direction of the gradients (oriented gradients) is used as features. For every pixel, the gradient has a magnitude and a direction. Equation (7) and Equation (8) shows the formula for finding magnitude and direction of the gradient where gx and gy are x and y component of the gradient.

$$g = \sqrt{gx^2 + gy^2} \tag{7}$$

$$\theta = \arctan(\frac{gy}{gx}) \tag{8}$$

The histogram is essentially a vector (or an array) of the number of bins corresponding to angles 0, 20, 40, 60, …, 360. The number of bins used in this thesis was nine. After creating histograms for all the micrographs, the ISE compared the histogram of an input micrograph with the histogram of all the micrographs in Dataset 1. The top five micrographs with least difference were selected, as visually similar micrographs. The chi-squared distance was used to compare the histogram. Equation (9) shows the formula for chi-squared distance and the value of ϵ used, in this thesis, was 2e-10.

$$d(x, y) = \frac{1}{2}\sum \frac{(x-y)^2}{(x+y+\epsilon)} \tag{9}$$

Figure 10 shows the image search result for one query micrograph, where the type of an alloy, of all the visually similar micrographs was predicted, by Model 4, to be of copper. In Figure 10, the first micrograph is the input micrograph.

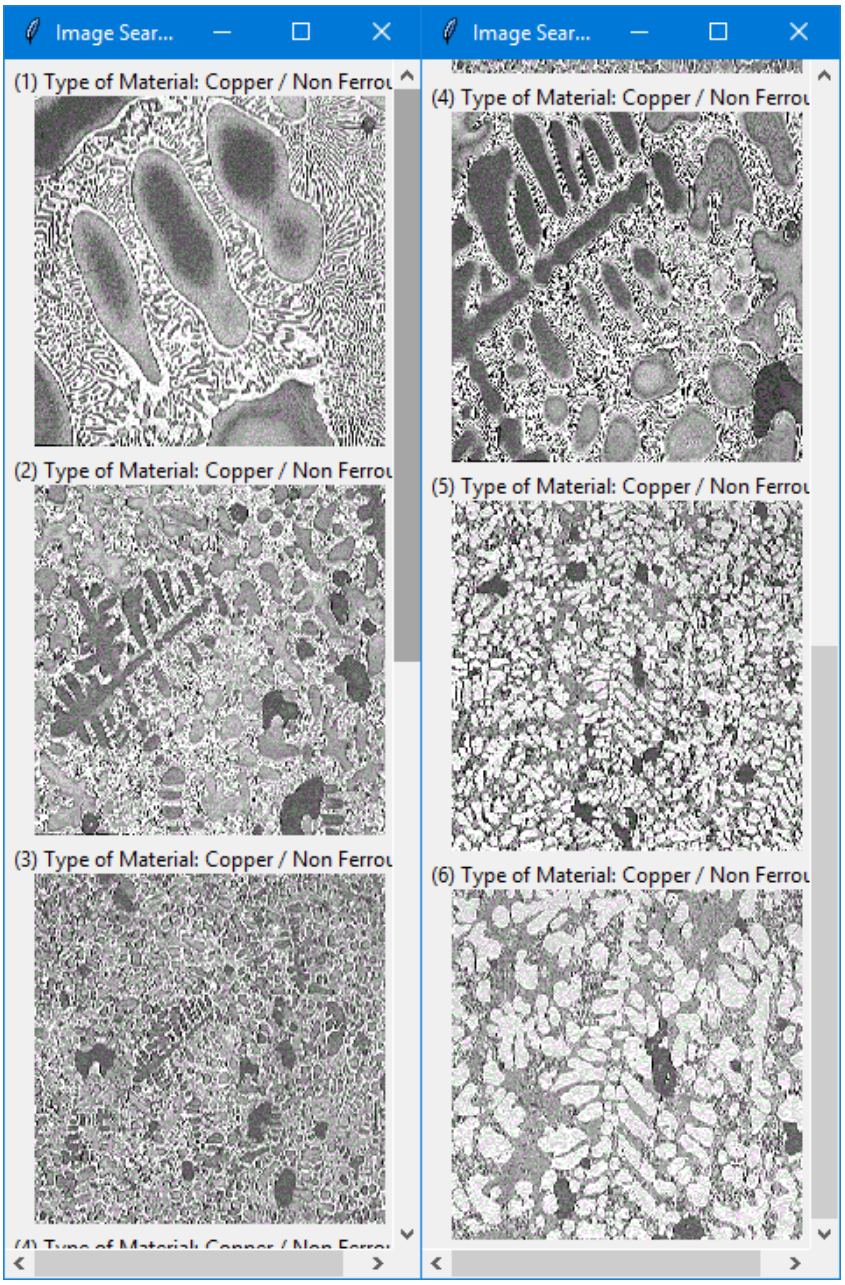

FIGURE 10. Visually similar micrograph with same alloy type.

Figure 11 shows the image search result for another micrograph, where the type of all visually similar micrographs, was predicted, by Model 4, to be of a different alloy type. In Figure 11, the input micrograph is of type steel, and for the search results, there are two micrographs of steel, one micrograph of Titanium, one micrograph of iron and one micrograph of a superalloy.

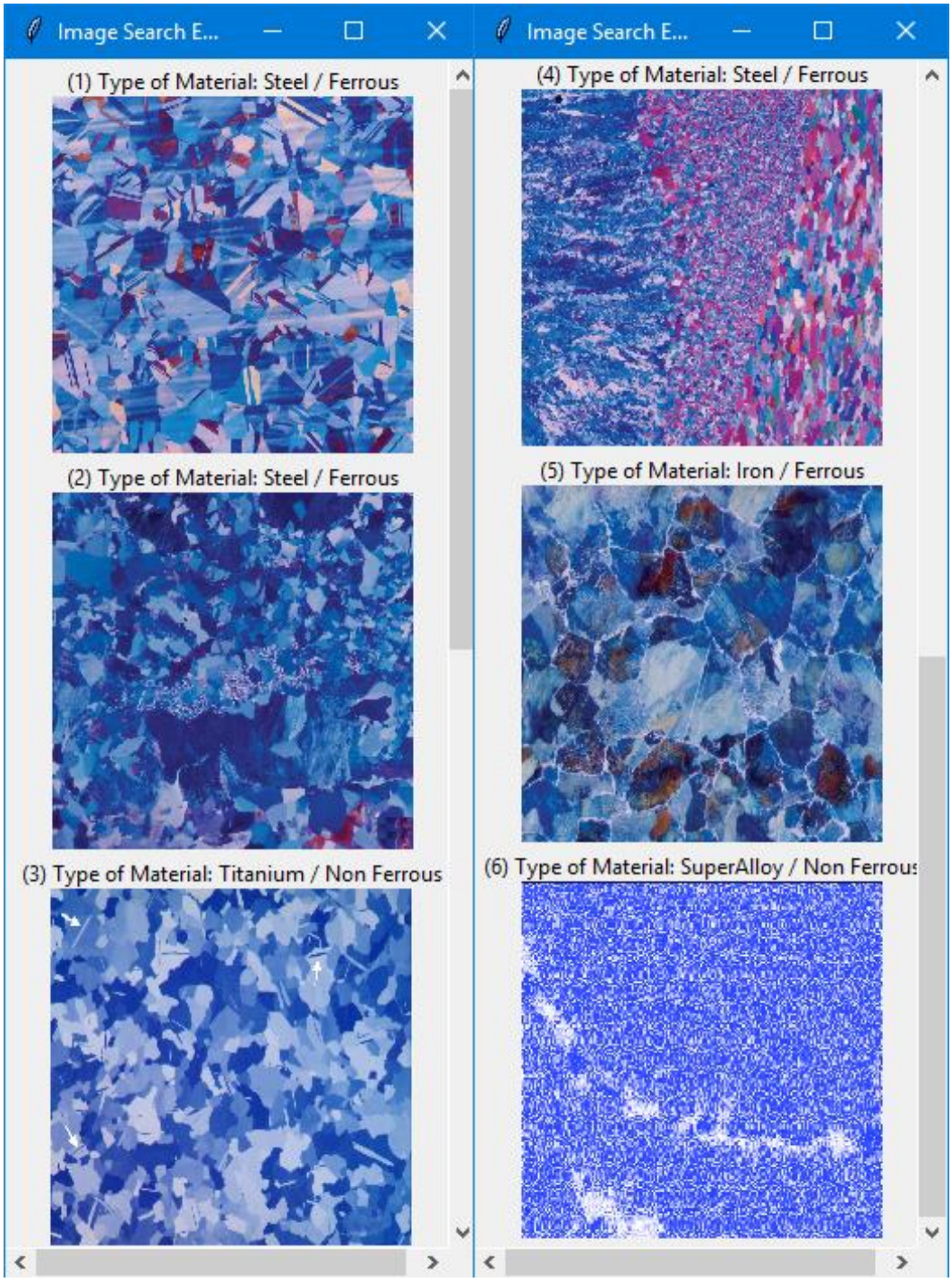

FIGURE 11. Visually similar micrograph with different alloy type.

CHAPTER 7

SOFTWARE AND HARDWARE SPECIFICATIONS

The code for developing and training all the neural network models and the code for building ISE was carried out using Python version 3.6 with the help of various open source libraries. The keras library was used for feature extraction and for creating the hidden layers. The sklearn library was used, for the training of all models. Python open cv2 library was used to construct histograms for HOG.

The hardware configuration used was, a virtual instance on the Google cloud platform with 2.2 GHz Intel Xeon E5 v4 (Broadwell) platform and four virtual CPU and 10GB memory.

CHAPTER 8

CONCLUSION AND FUTURE WORK

Different machine learning techniques were applied to build and train five different neural network models, which can automate the identification of alloy from its microstructure (Task 1) and classify that microstructure into Ferrous, Non-Ferrous, or Others class (Task 2). All the models were built using a different combination of feature extraction method, hidden layers, and classifiers. Results demonstrated that pre-trained ResNet50 architecture, when used as a feature extraction method was able to achieve an outstanding accuracy of 92.01% (±0.54%) and 94.31% (±0.59%) for Task 1 and Task 2 respectively.

ISE was successfully able to search for visually similar micrographs. The critical point to consider was that, even though the microstructures were visually identical, they were of a different kind of an alloy material. So, a human, who has no prior knowledge about the alloy microstructures would identify them as of the same type of alloy, but a trained neural network model was able to identify the difference in visually similar microstructures.

The source code for building and training Model 4 is on GitHub [11], and it also includes the trained model weights which be used in future to study more detailed features in microstructures.

Future Work

In this thesis, machine learning methods have given good results to identify the type of an alloy from its microstructure. New models can be built using Model 4, trained in this thesis, as a feature extractor. The new model can then be trained, on more diverse microstructure data set to learn better features of microstructures. These feature vectors can then be used to automatically

identify the type of composition of an alloy from its microstructures and identify the phase or condition under which the micrograph was captured.

REFERENCES

REFERENCES

[1] A. Chowdhury, E. Kautz, B. Yener, and D. Lewis, “Image Driven Machine Learning Methods for Microstructure Recognition,” *Computational Materials Science,* vol. 123, Oct. 2016, pp. 176-187.

[2] Metallurgy and Materials Engineering, “Liberty Ship Failures,” Dec. 2015; https://metallurgyandmaterials.wordpress.com

[3] Failure Knowledge Database 100 selected cases, “Brittle Fracture of Liberty Ships,” March 1943; http://www.sozogaku.com/fkd/en/lisen/hyakulisen.html

[4] B. DeCost and E. Holm, “A Computer Vision Approach for Automated Analysis and Classification of Microstructural Image Data,” *Computational Materials Science,* vol. 110, Aug. 2015, pp. 126-133.

[5] E. Alpaydin, “Dimensionality Reduction,” *Introduction to Machine Learning,* MIT Press, 2014, pp. 115-160

[6] A. Singh, N. Thakur, and A. Sharma, “A Review of Supervised Machine Learning Algorithms,” *Proc. Intl. Conf. on Computing for Sustainable Global Development (INDIACom),* 2016, pp. 1310-1315.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Adv. Neural Inf. Process. Syst.,* vol. 25, 2012, pp. 1097-1105.

[8] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” presented at the IEEE Conf. on Computer Vision and Pattern Recognition, 2016; DOI: 10.1109/CVPR.2016.90

[9] B. Du, W. Xiong, J. Wu, L. Zhang, L. Zhang, and D. Tao, “Stacked Convolutional Denoising Auto-Encoders for Feature Representation,” *IEEE Transactions on Cybernetics,* vol. 47, no. 4, Apr. 2017, pp. 1017-1027.

[10] D. P. Kingma and J. L. Ba, “Adam: A Method for Stochastic Optimization,” presented at the Intl. Conf. on Learning Representations, 2015.

[11] N. Shah, “Implementing Machine Learning Algorithms to Automate Identification of Material Microstructures,” Jun. 2018; https://github.com/Niyati1/Thesis-Machine-learning-algorithms-for-microstructure-of-materials.