ITC 3/49
Information Technology and Control
Vol. 49 / No. 3 / 2020
pp. 381-394
DOI 10.5755/j01.itc.49.3.26792

A Study of Supervised Combined Neural-Network-Based Ultrasonic Method for Reconstruction of the Spatial Distribution of Material Properties

Received 2020/06/16 | Accepted after revision 2020/07/13

http://dx.doi.org/10.5755/j01.itc.49.3.26792

HOW TO CITE: Dapkus, P., Mažeika, L. (2020). A Study of Supervised Combined Neural-Network-Based Ultrasonic Method for Reconstruction of the Spatial Distribution of Material Properties. *Information Technology and Control*, 49(3), 381-394. https://doi.org/10.5755/j01.itc.49.3.26792

A Study of Supervised Combined Neural-Network-Based Ultrasonic Method for Reconstruction of the Spatial Distribution of Material Properties

Paulius Dapkus*
Faculty of electronical and electronics engineering; Kaunas University of Technology; Studentu Str. 50, LT-51368, Kaunas, Lithuania; phone: +370 (37) 300 421; fax: +370 (37) 324 144; e-mail: paulius.dapkus@ktu.edu

Liudas Mažeika
Faculty of electronical and electronics engineering; Kaunas University of Technology; Studentu Str. 50, LT-51368, Kaunas, Lithuania; phone: +370 (685) 28 0 85; fax: +370 (37) 324 144; e-mail: liudas.mažeika@ktu.lt

*Corresponding author: paulius.dapkus@ktu.edu

This paper examines the performance of the commonly used neural-network-based classifiers for investigating such structural noise in metals as grain size estimation. It is extremely difficult to determine the grain size of objects only by the internal structure features of the object. When the structured data is obtained, a proposed feature extraction method is used to extract the feature of the object. Afterwards, the extracted features are used as the inputs for the classifiers. This research study is focused on using basic ultrasonic sensors to obtain object's structural grain size. The performance for the used neural-network-based classifier is evaluated based on recognition accuracy for an individual object. Furthermore, traditional neural networks, namely, convolutional and fully connected dense networks are shown as a result of the grain size estimation model. To evaluate

the robustness property of neural networks, the original samples data are mixed for three types of grain sizes. Experimental results show that combined convolutional and fully connected dense neural networks with classifiers outperform the other single neural networks with original samples with high signal-to-noise ratio data. The dense neural network as itself demonstrated the best robustness property when the object samples did not differ from trained datasets.

KEYWORDS: Material classification, Neural networks clusterization, ultrasonic non-destructive testing.

1. Introduction

Metal aging in most cases leads to changes in the structure of the materials such as the appearance of microcavities, cracks, which later can lead to the development of bigger damages of the structure, and as consequence, the materials lead to destruction. To examine the metals with non-destructive testing methods, the detection of the changes in noisy material internal structure is a very challenging task [2]. Even non-destructive assessment of the grain size inside the metals in principle is not solved. One of the most promising techniques for such a problem is ultrasonic measurements. This article presents a technique for non-destructive assessment of the grain size in metals based on the analysis of ultrasonic signals using neural networks (NN). The proposed NN methods are designed to learn the features of the object mechanical structure and assess the size of grains in different positions of the component under inspection.

The problem is to estimate grain size from the signal measured on a particular material. On the other hand, only destructive methods are currently used for grain size estimation and there are no non-destructive techniques that enable the assessment of the grain size inside a metal.

Some methodology for grain size estimation from structural noise in metals is called "Data-driven," and it extracts damage-sensitive features concentrating only on the unusual areas in sensor data instead of requiring a model of the structure. Some papers in this category operate on modern signal processing methods [1] and process a sudden change in signals caused by the presence of damage. In some ways, there is the research of [8] analyzing the characteristics of response signals under the wavelet transformation, and it was demonstrated that the wavelet coefficients can clearly show the moment when damage occurs. Another methodology in this category is when the damage detection task is treated as a pattern recognition problem based on a time series [15]. For detecting damage by modeling a dynamic signal using the autoregressive moving average (ARMA) model and observing ongoing changes in the model coefficient, a data-driven approach is proposed [3]. By using analysis of the moving base component analysis and robust regression analysis that detects damage based on statistical characteristics from time area data. However, the functions used in the above maintained studies are more oriented towards the detection of the damage with statistical methods.

The purpose of this research for grain size classification is to classify an unknown object into a predefined grain group consisting of pre-classified sets of objects with similar patterns to an unknown object. For many years, it was the artificial NN that was used to detect nature patterns [5]. In that manner, the NN is capable of detecting patterns with given features.

This is a very important NN research field that has a variety of areas such as health diagnostics, human resource optimization, language recognition, etc. In scientific literature, basic classification methods can be met as a logic-based approach. For reconstruction of spatial distribution of material properties, the decision graphs as a decision tree methods are much more effective and is known as subset of a logically based classification method and it is classified by sorting inputs according to characteristic property values. This methodology approach has shown better accuracy and clarity of the classification in images using the convolutional neural network [18]. The accuracy of the classification process can be improved by using the decision tree classifier.

Connecting multiple NN can further improve the performance of the overall decisions which NN is making. Both different processing methods such as convolutional NN and fully connected dense networks can be used for sampling the image kernels, ending with the

decision [14]. However, if both separate networks of dividing the decision make the wrong choice, it becomes more inaccurate than using one strong network with fine-tuned parameters [16]. For solving this problem multi-layered and combined NN are used [4].

One possible way of extracting features is by computing dynamic structural signatures or designing their derivative to gain greater sensitivity of damage. Damage detection can be done by monitoring ongoing changes in these signatures [17]. As a major characteristic, the natural amplitude changes are tested for convenience in the measurement. Some indirect damage detection studies exploring the detection of damage are based on the changes in the frequencies of the structure [12].

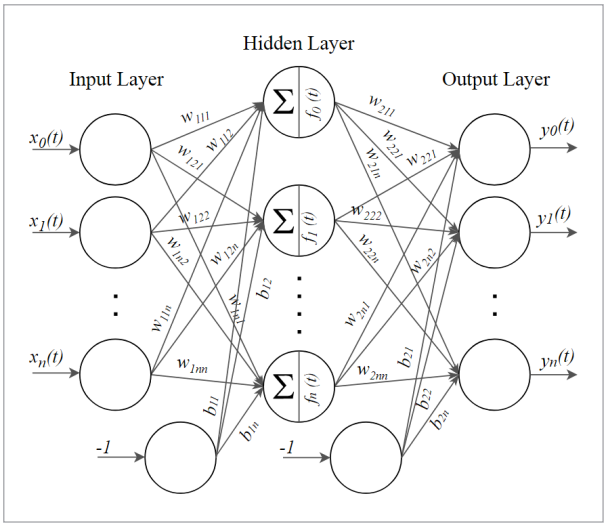
The article is laid out in three sections followed by conclusions. In the first section, the general approach is briefly described. The second section presents the model architecture, i.e. the structure of the NN. In the last section, the verification of the proposed model based on experimental measurements is presented.

2. The General Approach of the Method for Grain Size Estimation

The general approach is based on the application of fully connected dense NN. However, such a network can be used as a "post" classifier only, not for all the data processing [6]. The dense networks consist of several processing elements that are highly interconnected and transform a set of inputs into a set of desired outputs. The result of the transformation is determined by the element properties and weights associated with the interconnectors. By modifying the connections between the nodes, the network can adapt to the desired outputs.

A conventional network has an information forward structure, where the data flows forward from the network inputs through many hidden units and eventually to the output blocks. A typical densely connected network is shown in Figure 1, where the neurons or nodes are arranged in a discrete multilayer topology. These units serve to introduce the values of the input variables. The hidden and output layer neurons are each connected to all units in the previous layer.

Figure 1
Fully connected dense NN architecture



A fully connected dense NN with one input layer, n_l hidden layers, and one output layer is presented below in Figure 1. The network input with data index count as idx is represented by set of signals measured on the object under investigation $X(idx) = \left[x_1(idx)\, x_2(idx) \ldots\ldots\ x(idx)_{n_{input}}\right]$ and the NN output such as $Y(idx) = \left[y_1(idx)\, y_2(idx) \ldots\ldots\ y(idx)_{n_{output}}\right]$, where n_{input} is the number of input nodes in the input layer and n_{output} is the number of output nodes in the output layer. The output of the node in the input and hidden layers is represented as follows [11]:

$$f_i^{n_l}(t) = tf_l\left(\sum_{j=1}^{n_{n_h}^{(n_l-1)}} \omega_{ij}^{(n_l)} f_j^{(n_l-1)}(t) - b_j^{(n_l)}\right), \quad i = 1,2 \ldots., n_h^{(n_l)}, \tag{1}$$

where $f_i^{n_l}(t)$ is the transfer function; $n_h^{(n_l)}$ is the number of hidden nodes; $b_i^{(n_l)}$ is the bias in n_l-th hidden layer; $\omega_{ij}^{(n_l)}$ is the weight between j-th and i-th nodes of the hidden layers $n_{n_h}^{(n_l-1)}$ and $n_h^{(n_l)}$. The output of the NN is given as follows [11]:

$$y_i(t) = tf_{n_l+1}\left(\sum_{j=1}^{n_{n_h}^{(n_l)}} \omega_{ij}^{(n_l+1)} f_j^{(n_l)}(t) - b_j^{(n_l+1)}\right), \quad i = 1,2 \ldots., n_{out}. \tag{2}$$

Such a neural network cannot be directly applied for the estimation of the grain size at first because the accuracy of fully connected NN will be poor and will not be usable for this application [9]. To train this network, big data is required with multiple grain sizes labels. To overcome this problem, the novel combined neural network consisting of several NN as convolutional and fully connected dense networks was proposed. The method is based on an analysis of ultrasonic structural noise which is assumed to contain information about the size of the grain. The performance of the proposed method can be described in the following steps:

1 The ultrasonic signals are acquired on the object under investigation performing conventional B-scans. The time interval of the signals containing structural noise and back wall reflection is recorded and stored. The scanning and signals acquisition is performed using several excitation frequencies to cover wide band frequency ranges;

2 The B-scan images acquired at different frequencies are processed and converted into the format that is suitable for application input data for NN. In our case, B-scan images were combined into the pseudo or virtual RGB format. A more detailed description is presented in the following sections;

3 At the last step, the network is trained on the samples with known grain size. The result of such processing is the average size of the grain in NN input classifier grain size ranges.

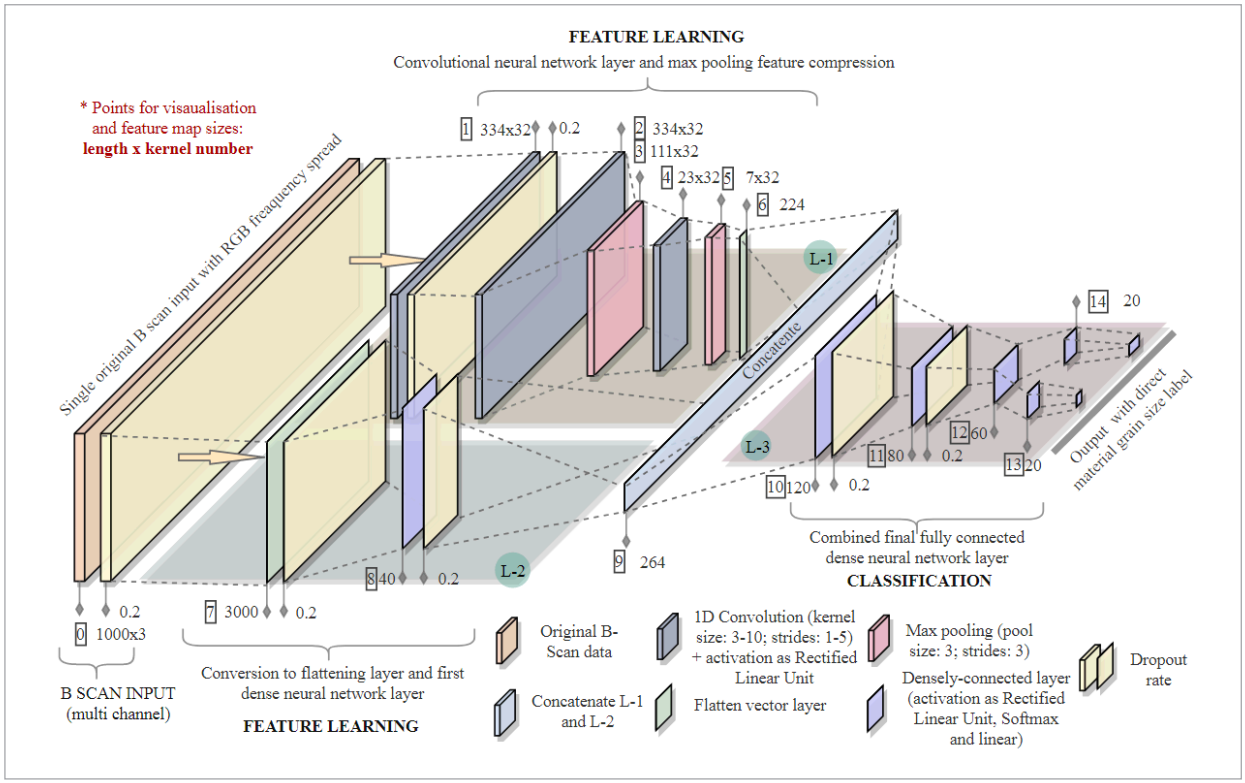
The key task is to develop such neural network architecture which can accept the B-scan signals as input data reliably to estimate the size of the grain.

3. Model Architecture and Methodology

To address structural noise issues, a novel NN architecture is designed. The proposed network can use full A-scan signals from three different frequency ultrasonic transducer data points with the length of 2000 elements. The NN model was created as shown in Figure 2 that the input layer needs three frequency

Figure 2

The main architecture of grain size in metals detection method shape. Using multiple spreaded neural networks - Convolutional neural networks and Dens fully connected neural networks



data vectors as the input layer. This process of different frequency sensor data as separate frequency channels is used to learn structural noise features, and to achieve direct grain size estimation simultaneously. Moreover, for NN to work it is important to prepare the sensors data. The preparation is done by converting the signals to positive values and combining them into three channels as one single dataset array. To address NN fast learning, a positive value conversion is required; in this way, the binary subtraction manipulation is avoided and the information of signal will not be lost.

The simplified flow chart of the proposed method is shown in Figure 3, which is organized as follows: (1-2-3) obtaining structural responses data from three different ultrasonic transducers – 2.25Mhz, 5Mhz, and 10Mhz. Mainly transducers are chosen to spread across from 2Mhz to 10Mhz frequency ranges.

The inner block contains data augmentation (Figure 3, block-4), an operation used to facilitate the performance of the NN by creating more data. The data are processed by preprocessing procedures which are combined to a single vector as shown in Equation 3 as a multi-frequency channel dataset (1-2-3 in Figure 3). This combination represents a different transducer and different frequency used to measure the object samples:

$$\mathbf{Input(idx)} = \left|tr_{f_0}\right| + \left|tr_{f_1}\right| + \cdots \left|tr_{f_n}\right|, \tag{3}$$

where *idx* represents A-scan signal iteration index and tr_{f_n} as single ultrasound transducer measured A-scan signal.

The second step is dataset labeling process which is done after three frequencies vector construction. The labeling is used to label the data as to direct grain sizes of three types. After dataset creation, the NN training is required.

Another important part is NN evaluation in the training process visualization of inner network convolutions (Figure 3, section-8). This block is used for tuning of NN interpretation of how the training is progressed at separate stages and to make it easier to calibrate the NN for future optimizations.

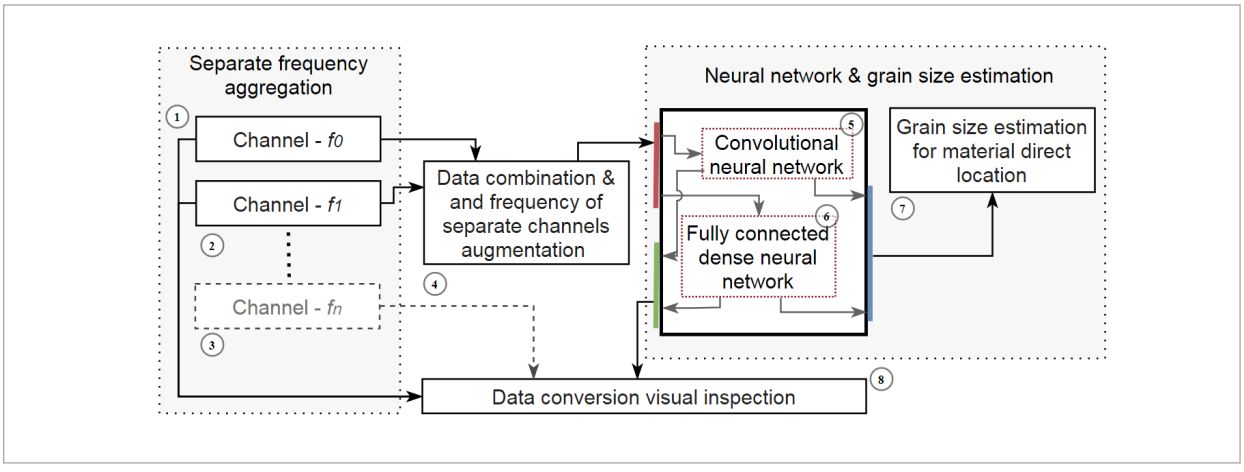
In this proposed NN method, the convolution and pooling are the most important and unique operations. Generally, convolution is an operation on two real-valued functions, such as [13]:

$$F(i) = \int_{-\infty}^{\infty} \mathrm{S}(n)\mathrm{K}(i-n)dn. \tag{4}$$

The Equation 4 is usually known as the input S and the kernel or filter K. Furthermore, the output here is referred to as a feature map or single neuron weight as the filter parameter. Supposedly that the integer domain has defined functions S and K, then the convolution can be called Equation 5, where *vs* and *vk* are maximally fit indexes in the S and K functions, respectively. In this case, it can be assumed that the index value is always zero. Convolution is commutative and the two representations in Equation 5 are equivalent. The first expression $\sum_{n=1}^{v_s} S(n)K(i-n)$ is more suitable for implementing a machine learning algorithm.

Figure 3

Estimating grain size in materials as separate channel frequencies. Multiple frequency addition to neural network and visualization of neural network inner block performance as for human eye interpretation



This is because kernel K, most often a multidimensional array of parameters, is much smaller in the NN context than input S. This yields a much lower maximum valid index obtained in the kernel. Moreover, in the latter term of Equation 5 the inputs S are folded relative to the kernel K. Assuming that as n increases, the index to kernel increases but the index at input S decreases; it brings the property to convolution.

$$F(i) = \sum_{n=1}^{v_s} S(n)K(i-n) = \sum_{n=1}^{v_k} S(i-n)K(n). \quad (5)$$

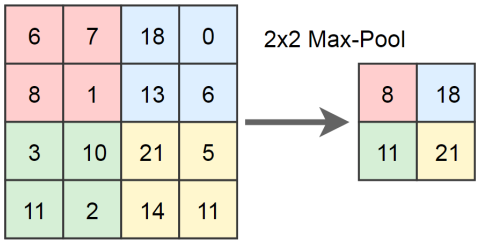
As can be seen in Equation 6, most machine learning libraries accept a version of convolution without flipping. Even though, in reality it should be called cross-correlation. Nevertheless, in the machine learning field, it is still called "convolution". This customization makes sense because the corresponding parameters in the kernel will be learned by the learning algorithm and it does not matter whether the input is inverted or not.

$$F(i) = \sum_{n=1}^{v_k} S(i+n)K(n). \quad (6)$$

Downsampling data strategy is the second important operation, since the maximum pooling action is part of a typical convolutional layer, as shown in Figure 4. It gives a statistical summary of the output elements that are nearby. One type of data pooling as a downsampling operation is the max-pooling. Just like with convolution, but in this case, instead of multiplying matrices, step by step it chooses the maximum within its "kernel size", otherwise known as pool length. Moreover, pool stride is the length of gaps between every two neighbors. Typically, it is set to the same length as the pool length (the value length of a post pooling process), while in convolution, a similar parameter is configured to be one. For example, stating that the output before the maximum pooling layer is a vector such as (1, 2, 3, 4) and the pool length and pool step are two, then the pooling exit will be (2).

Figure 4

Max pooling function with a sliding window (filter size) of 2×2. The simple maximum value is taken from each window to the output feature map



The pooling operation mainly brings two benefits: firstly, it helps to make the data representation more invariant to the small variance of the input. On account of the result of pooling being a statistical summary, a small variance in input may change its statistic characters slightly. It is a useful property to make a NN more robust to detect whether some feature is present or not. Secondly, the pooling operation reduces the size of the feature map, which is essential to improve the computational efficiency of the network.

Batch normalization is used and achieved through normalization steps. These steps fix the means and variances of each layer inputs. A normalization would be conducted over the entire data set [10]. In brief, the distribution of internal activations will continuously vary with the changes of network weights during training, which makes the learning algorithm to fit these unstable distributions in every training step, leading to a low convergence rate. During NN training, batch normalization is used as a batch mean Equation 7 and batch variance Equation 8:

$$\mu_D = \frac{1}{m}\sum_{i=1}^{m} x_i, \quad (7)$$

$$\sigma_D^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_D)^2, \quad (8)$$

then normalize the layer inputs using the previously calculated batch statistics as

$$xm_i = \frac{x_i - \mu_D}{\sqrt{\sigma_D^2 + \epsilon}}, \quad (9)$$

where ϵ [1, d] d-dimensional input. The next step is scaling and shifting to obtain the output of the layer as of

$$y_i = \gamma xm_i + \beta, \quad (10)$$

where the parameters γ and β are subsequently learned in the optimization process.

During training, for every batch of data, D = {x1,...,

xm}, the algorithm calculates the mean and variance, then shifts and scales the origin data to zero-mean and one variance. Finally, it introduces two learnable parameters, γ, and β, to hold the model flexibility. The batch normalization layer keeps its output following a similar distribution. As a result, the difficulty of training in the next layer is reduced, giving rise to fast convergence.

To classify input data, it is necessary to have a layer for predicting classes, which is usually located at the last layer of the convolutional NN architecture. The most prominent method to date is using the *softmax* function given by Equation 11 [7], which is expressed as the probabilistic expression for the *i-th* training example out of *n* number of training examples, the *j-th* class out of *n* number of classes, and weights *W*, where $W_n^T x^i$ are inputs of the *softmax* layer. The sum of the right-hand side for the *i-th* input always returns as 1, as the function always normalizes the distribution. In other words, Equation 11 returns probabilities of each input classes (*for i=1⋯m*)

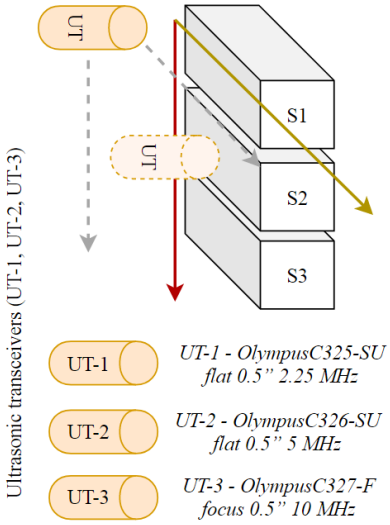
$$P\left(y^i = n \middle| x^i; W\right) = \begin{bmatrix} p\left(y^i = 1 \middle| x^i; W\right) \\ p\left(y^i = 2 \middle| x^i; W\right) \\ . \\ . \\ p\left(y^i = n \middle| x^i; W\right) \end{bmatrix} = \tag{11}$$

$$= \frac{1}{\sum_{j=1}^{n} e^{W_j^T x^i}} \begin{bmatrix} e^{W_1^T x^i} \\ e^{W_2^T x^i} \\ . \\ . \\ e^{W_n^T x^i} \end{bmatrix}.$$

4. Validation of the Proposed Model

For dataset collection, the experimental stand was created of three samples in Figure 6. The samples were stacked on each other for maximizing sample conjunction. In this way, collected signals were linear as possible and that helped for further signal manipulation. As shown in Figures 5 and 10, 5 to 2.5 MHz immersive ultrasonic transducers with a focus of 375mm were mounted on the Precision 6-11 axis TecScan system scanner. As for the scanning, all the measurements are done in the water tank, the ultrasonic waves from the transmission/reception (T/R) transducer were digitized and stored to a PC for later analysis and proposed NN training methods.

Figure 5
Data signals in multiple frequencies (UT-1; UT-2; UT-3) gathering method. All three types of material samples (S1-S2-S3) stacked on each other for gathered data linearity in the same conditions



All signals are picked in a full signal cycle from material bottom. To gather more accurate signals, all used signals were in an averaging mode as 64 counts. For gathering more signals from material samples, there were experiments in picking more signals from samples. In this way, the scanner ultrasound transducer movement path was dynamically changed not to overlap the signals in each other. Experiments showed that for this method for finding material grain size it is better to use a 1x1mm step. It is used to not overtrain the NN.

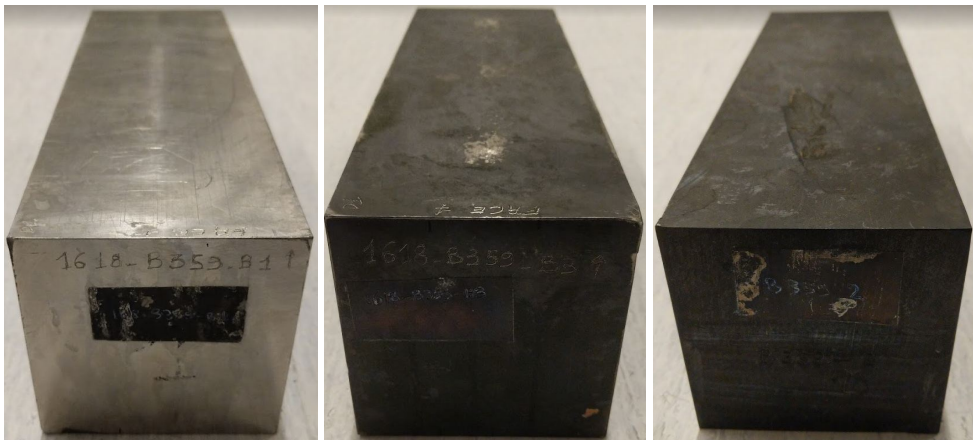
All signals use some gain to not overcome an 80% signal to window ratio. In that way, the first step is to inspect the process of B-scan and fine-tune the amplitude.

To maximize data usage for NN to learn, all the material samples defect holes are eliminated from the dataset.

All the measured samples as of A, B, and C are shown in Figure 6. There is a need to mention that sample C has unpredicted grain size spread in this sample, and it is considered as 400 μm.

Figure 6

Sample A from the left side with a grains average diameter ~80μm. Sample B with large grains average diameter ~800um. Sample C with intermediate grain size which average diameter is ~400μm. The materials mock-up is based on Ni alloy



The total number of raw data is 302400 different ultrasound signals. More briefly the total count on separate sample data is shown in Table 1. For training, the dataset was separated into two sets, one for training the NN – 70% (of all the measured A-B-C samples), and for model evaluation – 30%.

Table 1

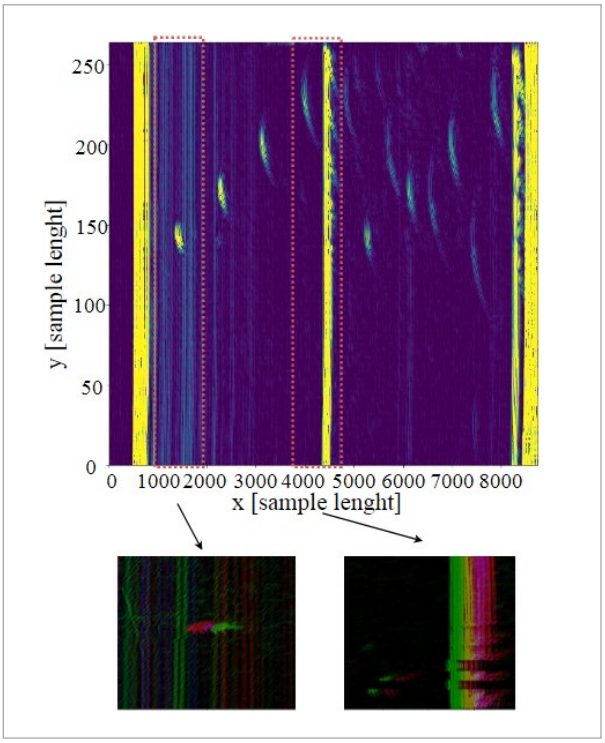
Dataset preparation in three different frequencies from 2.5 Mhz to 10 Mhz. Used a total of 302400 ultrasonic signals of two sides. All measures are done in full sample mirror

Sample	Sample side	f=2,5 Mhz	f=5 Mhz	f=10 Mhz	Total different signal count
A	Left, Right	12600	12600	12600	37800
B		12600	12600	12600	37800
C		25200	25200	25200	75600

All material scanning is stored in B-SCAN data. Later on, all scans are parsed and used as B-SCAN models. From Figure 7, it can be seen the extrapolated B-SCAN of material *sample A* with scanned holes. Moreover, the full output diagram is shown with the bottom signal which is in the middle and mirrored signal on the right side. In this example, 2.5 Mhz frequency is used.

Figure 7

At the top - initial one layer B-SCAN, 2.5 Mhz. At the bottom - trimmed (Red - first sample (A) 2.5 Mhz data spectrum, orange - first sample (A) 5 Mhz data spectrum, green - first sample (A) 10 Mhz data spectrum), combined with other frequencies and restored as RGB image



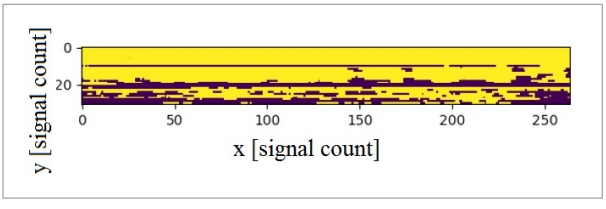
As mentioned above, the model can only accept data of the same form of the input layer. Therefore, data normalization is required. However, there is no reason for the data to be compressed to the same size for all data types of different sizes, as large samples would lose a large amount of information and no critical indicators are recognized, which form the ability of the NN to recognize the grain size of the material. To overcome this problem, data trimming is used, where only critical areas for sample identification are selected, as shown in Figure 10. Simplifying the filter with a common cooler is shown as follows.

For the next step, performing a search in all B-SCAN for the coordinates of the material bottom at each scanning point is needed. This process is started by moving further through a single-point ultrasound A-SCAN signal and looking for points that exceed a certain value. During experiments with data altering it was found that the exact value that best helps to find the average pixel index is 102, starting when the length of the sample is ⅓ and ending with ⅚. Once all the points are found, the first point is taken over 102 and its coordinate is entered into a two-dimensional array that preserves the reflection points of the entire sample. If this value is not found, then a value of 1 is given, indicating that the index was not found.

As shown in Figure 8, this search method is repeated with all 3 types of frequencies (*2.5 Mhz, 5 Mhz, 10 Mhz*). For detection of reflection, it is monitored whether the index values of a particular line in the array of bottom reflection indices are equal to *-1*. If this is the case, then the bottom reflection index is not found and the reflection index is searched in the next frequency. This methodology assumes that the material reflection line is found and does not deviate as the material is equal.

Figure 8

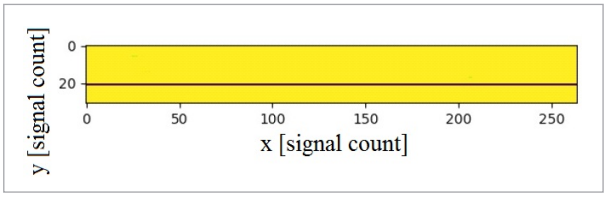
Result of searching material of bottom reflection for precise data cutting and preparing for NN input. Here, dark purple color represents where index in single A-SCAN data not found and yellow represents where bottom reflection is found



Performing material reflection search without auto-reflection (when reflection is not found) is shown in Figure 9. Here, the reflections from the bottom of the material are not evenly distributed everywhere, so the indexes that are not found are filled with support indexes from the same set of indexes that are found.

Figure 9

Result of adding support reflection indexes to form evenly distributed reflection dataset



After data trimming is done, both paths as object reflection start and reflection ending mirror data are combined to one. For combining the dataset, a python *Matplotlib* library is used to represent the result as a combined dataset. There, each frequency is represented as a separate channel, which allows us to get an understandable color image as RGB (red-green-blue) concept. As shown in Figure 7, all B-SCAN signal values are normalized – they are divided by *128*.

5. Model Training and Calibration

Conv1D layers were chosen as the basis of network architecture, which, like Conv2D layers (using a vector instead of 2D arrays only), automatically learn to identify specific structures. In this particular case, it is one of the essential characteristics of the network. This makes it possible to recognize the average size of the granules of the material. Three Conv1D layers are used in the last iteration of convolutional NN. The number of layers is chosen because the materials we want to classify are quite simple, given that they are rated by the network as 1.5D vectors rather than many attributes with 2D images. A higher number of Conv1D layers can provide more accuracy for the grid, but due to the good results of this particular model, a precise number of Conv1D layers were chosen.

Training settings are used on GPU (2x Nvidia 2080Ti) architecture for larger batch sizes which allows better utilization of GPU memory bandwidth and improved

computational throughput. For batch sizes, the fixed batch sizes are used as 512 and 4096, and NN maximum batch size as tested – 400 epochs. We selected rate as the learning rate with Adam optimizer, because the data set used is considered too small to be used for a lower learning rate. As activation layers, the ReLU is used for all layers, Softmax – for category extraction and linear activation is used for value extraction.

Table 2

Proposed NN all inner layers in the segmented layer approach. Starting from the input layer and finalizing with two fully connected dense networks for classification of grain size in the numeric dimension

Layer (type)	Layer output shape	Layer parameters	Layer connections
input_1 (InputLayer)	(None, 1000, 3)	0	None
dropout_1 (Dropout)	(None, 1000, 3)	0	input_1[0][0]
conv1d_1 (Conv1D)	(None, 334, 32)	320	dropout_1[0][0]
dropout_2 (Dropout)	(None, 334, 32)	0	conv1d_1[0][0]
conv1d_2 (Conv1D)	(None, 334, 32)	5152	dropout_2[0][0]
max_pooling1d_1 (MaxPooling1D)	(None, 111, 32)	0	conv1d_2[0][0]
dropout_3 (Dropout)	(None, 111, 32)	0	max_pooling1d_1[0][0]
conv1d_3 (Conv1D)	(None, 23, 32)	10272	dropout_3[0][0]
flatten_2 (Flatten)	(None, 3000)	0	dropout_1[0][0]
max_pooling1d_2 (MaxPooling1D)	(None, 7, 32)	0	conv1d_3[0][0]
dropout_5 (Dropout)	(None, 3000)	0	flatten_2[0][0]
dropout_4 (Dropout)	(None, 7, 32)	0	max_pooling1d_2[0][0]
dense_1 (Dense)	(None, 40)	120040	dropout_5[0][0]
flatten_1 (Flatten)	(None, 224)	0	dropout_4[0][0]
dropout_6 (Dropout)	(None, 40)	0	dense_1[0][0]
concatenate_1 (Concatenate)	(None, 264)	0	flatten_1[0][0], dropout_6[0][0]
dense_2 (Dense)	(None, 120)	31800	concatenate_1[0][0]
dropout_7 (Dropout)	(None, 120)	0	dense_2[0][0]
dense_3 (Dense)	(None, 80)	9680	dropout_7[0][0]
dropout_8 (Dropout)	(None, 80)	0	dense_3[0][0]
dense_4 (Dense)	(None, 60)	4860	dropout_8[0][0]
dense_5 (Dense)	(None, 20)	1220	dense_4[0][0]
dense_7 (Dense)	(None, 20)	1220	dense_4[0][0]
dense_6 (Dense)	(None, 3)	63	dense_5[0][0]
dense_8 (Dense)	(None, 1)	21	dense_7[0][0]

The study was conducted with or without convolutional layers which demonstrates that the use of convolutional layers has some drawbacks that prevent a certain level of abstraction that simplifies the training of the network. It has also been observed that the network is less prone to learning when expanding the convolutional layers and the high degree of accuracy leads to less accuracy in the evaluation of unseen data. For these reasons, a higher number of *Conv1D* layers is useful because it also allows better classification of unseen data, even using a lesser amount of data for network training.

According to Table 2 and Figure 2, in the depicted network architecture, it can be seen how the first convolutional layers identify and reduce the plethora of excess information for subsequent Dense layers. Due to this methodology, the number of network parameters is reduced from ~ 800 thousand to ~ 330 thousand parameters and allows to increase network training and material evaluation speed up to 1.4 times compared to a network without *Conv1D* layers. The data processing process is shown after Conv1d layers in Figure 10, where each layer is responsible for detecting specific properties and transmitting their abstraction to a deeper layer. Here, we can see the main activations in separate samples.

After the classification, the mean accuracy of the signal classification is ~ 93% and is changed as a significant difference in accuracy between the different classes of samples. After an analysis of different categories of single signal identification, sample A achieved the highest accuracy with ~ 98% of the accurate estimates, but samples B and C lag behind with ~ 88% and 92% accuracy, respectively. The phenomenon was studied in Figure 10, where it can be seen that after 3 convolutional layers and samples A and B show significant pattern differences compared to C, resulting in inaccuracy in the identification of the sample C between the categories. The difference between samples A, B and C is shown in Figure 10.

Even though, the experiments show network was with 93.4% grain size estimation probability, it can be assumed that excluding cases where the network is not convinced would result in even greater accuracy with this particular data. Therefore, this would lead to greater precision in this case.

Table 3 shows a comparison between the multiple models after networks have been trained. Here, AOCB is applying only one category sample. In the precision analysis of the different networks, the training was carried out by assigning 5000 signals per category of samples (in the case of AOCB - only one category sample). An accuracy of 93.4% was achieved with training validation data that were separated from the total training data with a coefficient of 0.2. Then there was a test with other data, assigning a different number of signals per assessment. The correct evaluation is successful if the vast majority of data signals have been correctly eval-

Figure 10

Activation of convolutional network layers. At the top are the averages of the random values of 100 signal units, at the bottom are the averages of 100 alarm activation values of 3 convolution layers. The picture highlights activation zones specific to a specific sample class. Samples: A – 80 µm, B – 400 µm, C – 800 µm

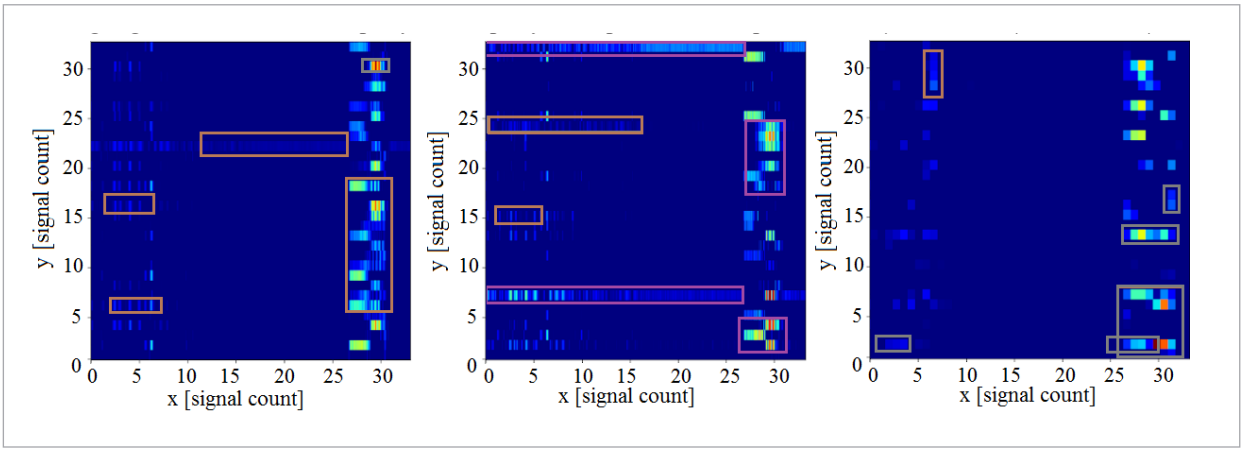


Table 3

Proposed models for different learning epoch iteration with each model accuracy predicting grain size

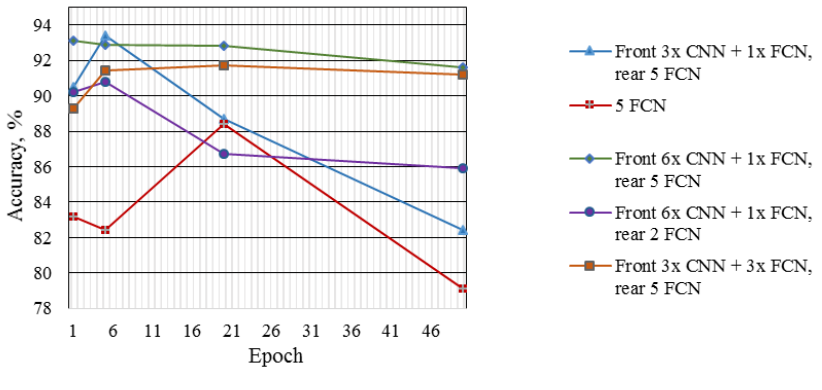
Model	Accuracy after learning iteration, epoch			
	1	5	20	50
3 Convolutional and 1 fully connected dense layers + rear 5 fully connected dense layers with 184648 parameters	90,5	**93,4**	88,7	82,4
3 Convolutional and 1 fully connected dense layers + rear 5 fully connected dense layers with 184648 parameters, AOCB*	73,5	**90,3**	81,3	76,6
5 Fully connected dense layers with 261654 parameters	83,2	82,4	**88,4**	79,1
5 Fully connected dense layers with 261654 parameters, AOCB*	82,3	81,3	**86,3**	77,9
6 Convolutional and 1 fully connected dense layers + rear 5 fully connected dense layers with 332360 parameters	**93,1**	92,9	92,8	91,6
6 Convolutional and 1 fully connected dense layers + rear 5 fully connected dense layers with 332360 parameters, AOCB*	**90,1**	90,1	90	89,8
6 Convolutional and 1 fully connected dense layers + rear 2 fully connected dense layers with 281240 parameters	90,2	**90,8**	86,7	85,9
6 Convolutional and 1 fully connected dense layers + rear 2 fully connected dense layers with 281240 parameters, AOCB*	84,2	**86,4**	84,8	84,1
3 Convolutional and 3 fully connected dense layers + rear 5 fully connected dense layers with 332360 parameters	89,3	91,4	**91,7**	91,2
3 Convolutional and 3 fully connected dense layers + rear 5 fully connected dense layers with 332360 parameters, AOCB*	69,5	87,9	**89**	85,5

uated. This is done 300 times for each sample and the overall percentage of the correctness of the estimates is derived. As the accuracy of each category is s > 50%, increasing the number of signals increases the overall accuracy until it reaches the 100% estimation. However, given that the training and testing samples were relatively similar and the number of tests carried out was limited, the best model was reported.

As of seen in Figure 11, the standard FCN network performance with data preparation is weakest of all (combined) tested. This is because the network consists of only dense layers.

Figure 11

Tested models accuracy predicting grain size for different learning epoch iteration with different model structures



6. Discussion and Conclusion

Results are obtained with the proposed model shows its 93.4% accuracy, even though the dataset is noisy. The proposed fine-tuned model observed in the multiple grain size estimations. Moreover, combined NN requires substantially fewer parameters and less computation to achieve state-of-the-art performances. Because we adopted structural noise for residual networks in our study, we believe that further gains in the accuracy of the proposed NN may be obtained by more detailed tuning of dataset and learning rate schedules.

Whilst following the NN combination rule, NN naturally integrates the properties of grain size identity mappings, deep supervision, and diversified depth from separate convolutions. They allow feature reuse throughout the layers and can consequently learn more compact data and, according to our experiments, more accurate models. Following that, this NN may be a good feature extractor for various computer vision tasks that need to extract features from noisy data. We plan to study such feature transfer with proposed NN in future work.

References

1. Amezquita-Sanchez, J., Adeli, H. Feature Extraction and Classification Techniques for Health Monitoring of Structures, Scientia Iranica. Transaction A: Civil Engineering, 2015, 22(6), 1931-1940.
2. Bai, L., Velichko, A., Drinkwater, B. W. Grain Scattering Noise Modeling and Its Use in the Detection and Characterization of Defects Using Ultrasonic Arrays. IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, 2019, 66(11), 1798-1813. https://doi.org/10.1109/TUFFC.2019.2927439
3. Cavadas, F., Smith, I. F. C., Figureueiras, J. Damage Detection Using Data-Driven Methods Applied to Moving-Load Responses. Mechanical Systems and Signal Processing, 2013, 39(1-2), 409-425. https://doi.org/10.1016/j.ymssp.2013.02.019
4. Cha, Y. J., Choi, W., Buyukozturk, O. Deep Learning-Based Crack Damage Detection Using Convolutional Neural Network. Computer-Aided Civil and Infrastructure Engineering, 2017, 32(3), 2013-14. https://doi.org/10.1111/mice.12263
5. De Fenza, A., Sorrentino, A., Vitiello, P. Application of Artificial Neural Networks and Probability Ellipse Methods for Damage Detection Using Lamb Waves. Composite Structures, 2015, 133, 390-403. https://doi.org/10.1016/j.compstruct.2015.07.089
6. Dung, C. V., Ahn, L. D. Autonomous Concrete Crack Detection Using Deep Fully Convolutional Neural Network. Automation in Construction, 2019, 99, 52-58. https://doi.org/10.1016/j.autcon.2018.11.028
7. Gao, B., Pavel, L. On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning. arXiv Preprint, 2017. arXiv.org > cs > arXiv:1704.00805
8. Hou, Z. K., Noori, M. N., Amand, R. S. Wavelet Based Approach for Structural Damage Detection. Journal of Engineering Mechanics, 2000, 126(7), 677-683. https://doi.org/10.1061/(ASCE)0733-9399(2000)126:7(677)
9. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K. Q. Densely Connected Convolutional Networks. The IEEE Conference on Computer Vision and Pattern Recognition, 2017, 4700-4708. https://doi.org/10.1109/CVPR.2017.243
10. Ioffe, S., Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv Preprint, 2015. arXiv.org > cs > arXiv:1502.03167
11. Le, T. N., Sugimoto, A. Deeply Supervised 3d Recurrent FCN for Salient Object Detection in Videos. Conference: The 28th British Machine Vision Conference (BMVC 2017), 2017, 38.1-38.13. https://doi.org/10.5244/C.31.38
12. Mohan, A, Poobal, S. Crack Detection Using Image Processing: A Critical Review and Analysis. Alexandria Engineering Journal, 2018, 57(2) 787-798. https://doi.org/10.1016/j.aej.2017.01.020
13. Munir, N., Kim, H. J., Park, J., Song, S. J., Kang, S. S. Convolutional Neural Network for Ultrasonic Weldment Flaw Classification in Noisy Conditions. Ultrasonics, 2019, 94, 74-81. https://doi.org/10.1016/j.ultras.2018.12.001
14. Simonyan, K., Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv Preprint, 2014. arXiv.org > cs > arXiv:1409.1556
15. Sohn, H., Farrar, C. R. Damage Diagnosis Using Time Series Analysis of Vibration Signals. Smart Materials

and Structures, 2001, 10(3), 446-451. https://doi.org/10.1088/0964-1726/10/3/304

16. Wu, R., Yan, S., Shan, Y., Dang, Q., Sun, G. Deep Image: Scaling up Image Recognition, arXiv Preprint, 2015, 7(8). arXiv.org > cs > arXiv:1501.02876
17. Young-Jin, C., Oral, B. Structural Damage Detection Using Modal Strain Energy and Hybrid Multi Objective Optimization, Computer-Aided Civil and Infrastructure Engineering, 2015,30(5), 347-358. https://doi.org/10.1111/mice.12122
18. Zheng, Q., Yang, M., Tian, X., Jiang, N., Wang, D. A Full Stage Data Augmentation Method in Deep Convolutional Neural Network for Natural Image Classification. Discrete Dynamics in Nature and Society, 2020, 2020(2), 1-11. https://doi.org/10.1155/2020/4706576