Ultrasonics 94 (2019) 74–81

Contents lists available at ScienceDirect

Ultrasonics

journal homepage: www.elsevier.com/locate/ultras

# Convolutional neural network for ultrasonic weldment flaw classification in noisy conditions

Nauman Munir[a], Hak-Joon Kim[a,*], Jinhyun Park[a], Sung-Jin Song[a], Sung-Sik Kang[b]

[a] Department of Mechanical Engineering, Sungkyunkwan University, Suwon 16419, Republic of Korea
[b] Korea Institute of Nuclear Safety, Daejeon 34142, Republic of Korea

ARTICLE INFO

*Keywords:*
Convolutional neural network (CNN)
Signal to noise ratio (SNR)
Ultrasonic testing
Weldment flaw classification

ABSTRACT

Ultrasonic flaw classification in weldment is an active area of research and many artificial intelligence approaches have been applied to automate this process. However, in the industrial applications, the ultrasonic flaw signals are not noise free and automatic intelligent defect classification algorithms show relatively low classification performance. In addition, most of the algorithms require some statistical or signal processing techniques to extract some features from signals in order to make classification easier. In this article, the convolutional neural network (CNN) is applied to noisy ultrasonic signatures to improve classification performance of weldment defects and applicability. The result shows that CNN is robust, does not require specific feature extraction methods and give considerable high defect classification accuracies even for noisy signals.

## 1. Introduction

Ultrasonic testing for detection and evaluation of defects in weldment is common and widely applied inspection method. These techniques use the shape and time of flight information of the reflected ultrasonic signals from defects. The major purpose of ultrasonic inspection is to detect, classify and characterize defects in weldment accurately and quickly. However, these techniques are error prone and require an experienced operator to correctly interpret the ultrasonic signatures of weldment defects. So, to alleviate the interpretation burden from operator, automatic ultrasonic defect classification system is the imminent need of industries.

Several artificial intelligence techniques have evolved during the last three decades to automatically classify weldment defects. These include ultrasonic pattern recognition techniques and neural networks, etc. Neural networks for ultrasonic flaw classification have been very popular in the past and several researchers have applied them for classification. Song et al. [1] classified the weldment flaws using probabilistic neural network by extracting time domain features from their ultrasonic scattering signatures. Cracks, slag and porosity defects were automatically classified by employing neural network on feature values extracted from ultrasonic signals using Fisher discriminant analysis [2]. Margrave et al. [3] experimented with different types and configuration of the neural network to find flaws in steel pipes using ultrasonic signatures. Liu et al. [4] classified and identified the type, location and length of cracks by using the neural network on characteristic values extracted from ultrasonic signals. Drai et al. [5] classified volumetric and planar defects by employing neural network on feature values extracted in the time domain, frequency domain and discrete wavelet representations. Song et al. [6] developed Intelligent Ultrasonic Evaluation System (IUES) for weldment flaw classification in a real-time fashion. Feed forward neural network along with wavelet e blind separation was exercised to classify positions, width and depth of defects in not accessible pipes [7]. Sambath et al. [8] applied the wavelet transform to collect 2D information of various types of defects and then utilized that data to improve automatic ultrasonic flaw detection and classification accuracies using the neural network. An embedded electronic system for decision support was developed to classify delamination and fractures in composites [9]. A comparison of different feature extraction and feature selection approaches was debated in [10] to classify lack of penetration, porosity and slag inclusion defects by the neural network.

Almost all of the above-mentioned approaches, except an effort made in [3], although have very good performance, however, very complex and may not practical for industrial usage. The main complexity in their implementation is the extraction of features by statistical or signal processing techniques. In spite of the fact, feature extraction techniques are very effective for dimensionality reduction but are exhaustive and require careful selection of features that remain insensitive to operating conditions.

* Corresponding author.
*E-mail address:* hjkim21c@skku.edu (H.-J. Kim).

https://doi.org/10.1016/j.ultras.2018.12.001
Received 16 August 2018; Received in revised form 24 November 2018; Accepted 1 December 2018
Available online 01 December 2018

Recently, due to the incredible increase in computing power, capable of handling large dimensional data, a fully connected deep neural network (DNN) with drop out was reported from the same platform [11] to classify cracks, lack of fusion (LOF), lack of penetration (LOP), porosity and slag inclusion defects without extracting any features from mixed frequency ultrasonic signals. Although, fully connected DNN performed reasonable well to classify weldment defects, however, the performance was checked for almost noise free ultrasonic signatures and noise is inevitable in industrial environments. Consequently, an extension to the previous work, this research is a step further to suit industrial needs for automatic weldments flaw classification in feature independent and noisy conditions.

In order to check the performance of fully connected DNN, the database used in previous works [6,11], was first augmented by the time shifting of signals as data augmentation is a quite effective technique to increase the amount of data for deep neural networks [12–15] to improve their performances. The additive white Gaussian noise (AWGN) was then added to the database at five signal to noise ratios (SNR) levels i.e. SNR 5 (high noise), SNR 7, SNR 10, SNR 15 and SNR 20 (low noise). The performance of fully connected DNN was then evaluated at each noise level. The result shows that DNN performed fairly well for low noise signals, however, for high noise signals, there is considerable decrease in its accuracies especially for cracks and slag defects.

Due to relatively low performance of fully connected DNN at high SNRs, a new neural network architecture with convolution layers was adopted, commonly known as the convolutional neural network (CNN). CNN firstly proposed by LeCun [16], later on, gave state of the art performances in the field of computer vision and speech recognition [17,18]. CNN has also been employed successfully to classify void and delamination defects in CFRP specimens [19] and for monitoring machine health [20]. So, in this study, we focused on the use of the convolutional neural network for noisy ultrasonic signals to classify weldment defects. The results demonstrate that the adopted convolutional neural network architecture is feature independent and works considerable well than fully connected DNN even for high noise signals.

## 2. Ultrasonic flaw database

In the previous work [6,11], a database of 720 almost noise free signals (original database) of deliberate welding defects i.e. cracks, lack of fusion (LOF), lack of penetration (LOP), porosity and slag inclusions from Flawtech specimens was used. The database signals were collected by 2 MHz and 4 MHz transducers. The original database contains 360 waves of 2 MHz transducer and 360 waves of 4 MHz transducer. The number of signals for each defect type are present in Table 1.

As the number of signals in the original database were very few and almost noise free, hence, in the current research, the size of original database was first artificially augmented by time shifting to increase the number of signals for each defect type. This augmented database was then altered with additive white Gaussian noise to make the signals comparable to field applicable signals. The details of augmentation and noise addition are as follows.

**Table 1**
Complete original database with number of defects for each signal type.

| Defects | No of Signals | | |
|---|---|---|---|
| | 2 MHz | 4 MHz | Total |
| Cracks | 120 | 120 | 240 |
| Lack of Fusion | 115 | 115 | 230 |
| Lack of Penetration | 35 | 35 | 70 |
| Porosity | 35 | 35 | 70 |
| Slag Inclusion | 55 | 55 | 110 |
| Total | 360 | 360 | 720 |

### 2.1. Data augmentation

Data augmentation is quite an effective method to increase the amount of data for deep neural networks [12–15] in the field of computer vision and artificial intelligence. In this technique, new instances are generated from existing ones that are similar to the original ones, thus the size of the database is artificially boosted. In the current approach, as there were very few examples in the original database to teach the neural network about the weldment defect classification in noisy conditions, the data is augmented to improve the prediction ability of neural networks. The technique used for data augmentation was the time shifting of the defect signals. This technique, in the terms of ultrasonic non – destructive evaluation, is analogous to changing the distance between the transducer and defect location. Therefore, to augment the data, signals were time-shifted 5 μs and 10 μs forward and backward as shown in Fig. 1.

Using the data augmentation method, the number of signals in the database were increased from 720 signals to 3600 signals to make an augmented database as shown in Table 2 and the number of signals for each defect in the augmented database are present in Table 3. So, augmented database has 3600 signals where each signal has 2048 sampling points.

### 2.2. Addition of noise

To include the effects of random processes in these signals, the signals in augmented database were altered with additive white Gaussian noise. Here, white Gaussian noise is used to represent electronic noise present in ultrasonic systems. The reason of assuming electronic noise as white Gaussian is due to the fact that any resistor produce thermal noise at non zero temperature. This rise in temperature mobilize the charges (electrons) inside the resistor, the sum of which can be measured as a voltage at the resistor terminals. Then owing to the central limit theorem, the aggregated voltage can be modeled as white Gaussian noise [21]. The complete description and mathematical details of white Gaussian noise are present in [21,22].

There were 5 different noise levels selected for signal's modification, i.e. SNR 5, SNR 7, SNR 10, SNR 15 and SNR 20 where SNR 5 is the highest level of noise and SNR 20 is the low level of noise. In this manner, 5 versions of augmented database were synthesized i.e. augmented database with SNR 5, SNR 7, SNR 10, SNR 15 and SNR 20 each containing 3600 signals and have 2048 sampling points. The original signal and signal with SNR 5 noise level are shown in Fig. 2.

### 2.3. Training and testing datasets

Training the neural network on a given data set and checking its performance on the same data set is methodologically wrong because it can achieve high accuracy on the seen examples but may fail to predict the unseen examples. In order to investigate the actual performance of the neural network, it is common practice to divide the data into training and testing datasets. The network is trained on the training dataset while its accuracy is checked on the testing dataset. Therefore, augmented database for each noise level was divided into training and testing datasets.

However, as shown in Table 3, data in augmented database is imbalance, i.e. certain classes have more signals while others have fewer signals. Due to this imbalance, data in the augmented database for each noise level was divided into training and testing datasets in such a way that the relative frequency of each defect class remains preserved i.e. 90% for training and 10% for testing. Table 4 shows the number of signals for each defect in augmented database for training and testing datasets.

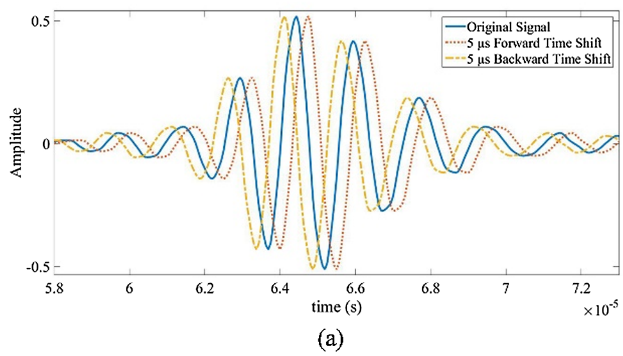


**Fig. 1.** Time shifting of signals. (a) 5 µs forward and backward, (b) 10 µs forward and backward.

**Table 2**
Augmented database synthesized by time shifting of signals.

| Database | | Shifting Time | No of signals |
|---|---|---|---|
| 1 | Original Database | No Shifting | 720 |
| 2 | Original Database | 5 µs forward shifting | 720 |
| 3 | Original Database | 5 µs backward shifting | 720 |
| 4 | Original Database | 10 µs forward shifting | 720 |
| 5 | Original Database | 10 µs backward shifting | 720 |
| Total (Augmented Database) | | | 3600 |

**Table 3**
No of signals for each defect in augmented database.

| Defects | No of signals |
|---|---|
| Cracks | 1200 |
| Lack of Fusion | 1150 |
| Lack of Penetration | 350 |
| Porosity | 350 |
| Slag Inclusion | 550 |
| Total | 3600 |

### *2.4. Scaling of datasets*

All the signals before feeding to neural networks were scaled with Eq. (1). This scaling was applied to get all the signals on the same scale and to avoid activation function saturation [23].

$$Scale = \frac{|Signal|}{max\ |Signal|} \tag{1}$$

## 3. Artificial neural network

Neural networks are mathematical models that are based upon the Rosenblatt's perceptron training algorithm that is vaguely inspired by biological neurons. Simplest feed forward fully connected neural network has three layers named as an input layer, a hidden layer and an output layer. The layers are connected in such a way that every node in the current layers is connected to each node in the previous layer and their connection has specific weight. A DNN has more than one hidden layer between input and output layers.

### *3.1. Convolutional neural network*

The convolutional neural network is a type of deep neural network that has convolutional layers as well as fully connected layers. Convolution is a mathematical function that is widely applied in the signal processing domain. In the convolutional neural network, convolutional layers actually utilize cross-correlation technique that is technically very similar to convolution [24].

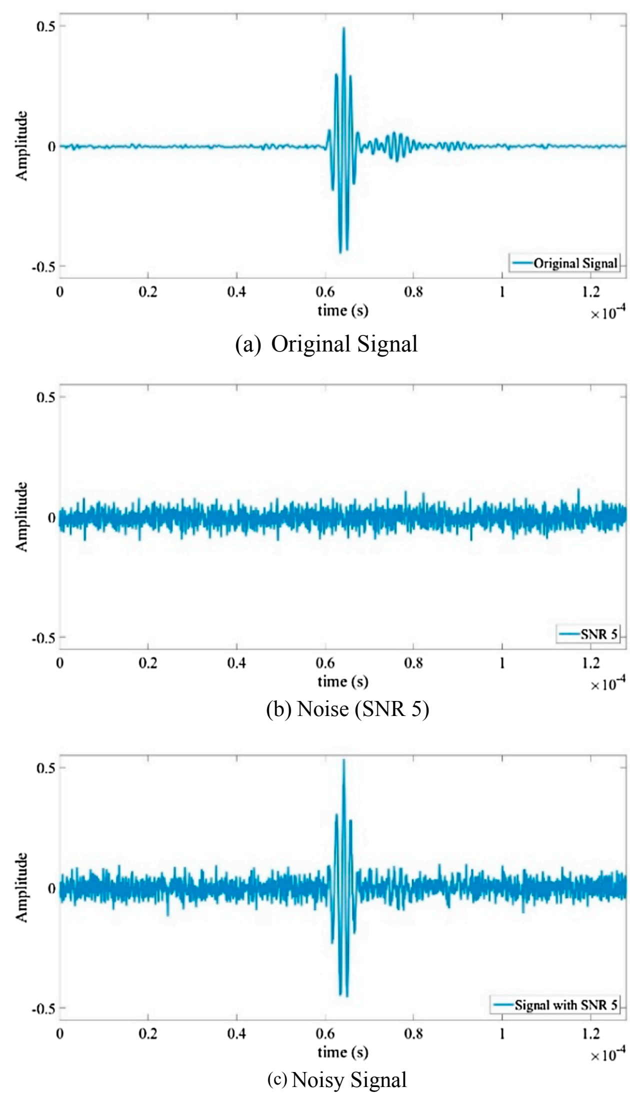


**Fig. 2.** Signal with SNR 5.

There are two important layers in CNNs i.e. feature extraction layers and classification layers. Convolutional layer and pooling layers are features extracting layers while classification layers are fully connected layers. Convolutional layers, unlike fully connected layers, not connected to every node in the input layer, but to specific local regions based upon the defined filters/convolutional kernels. This architecture allows the network to concentrate on low-level features, those are then assembled into high-level features. CNN also has ability to learn pattern

**Table 4**
Training and testing datasets for augmented database.

| Defects | Augmented Database *(SNR 5, SNR 7, SNR 10, SNR 15, SNR 20)* | |
|---|---|---|
| | Training Dataset | Testing Dataset |
| Cracks | 1080 | 120 |
| Lack of Fusion | 1035 | 115 |
| Lack of Penetration | 315 | 35 |
| Porosity | 315 | 35 |
| Slag Inclusion | 495 | 55 |
| Total | 3240 | 360 |

at one location and then determine it in some other location. This, in fact, is possible due to the same parameter sharing in filters. Pooling layer in the neural network is used to subsample the input that helps to reduce the computational load and also helps to avoid overfitting [24] however, increase complexity [25].

### 3.2. Fully connected deep neural network architecture

In the present work, the adopted fully connected deep neural network was designed in Tensorflow (*Google open source software for Deep Learning*). It contained one input layer, 3 hidden layers and one output layer. The number of nodes in the input layer were equal to the sampling points of the ultrasonic signals and there were 5 nodes in the output layer to classify 5 defects. The number of nodes in the hidden layers were decided after several trail runs and chose the number of nodes that gave best performance. The activation function used in the fully connected layers was rectified linear unit (Relu) $Relu(x) = \max(0, x)$ due to its state of the art performance in deep learning [26–28] and error was calculated with sparse softmax cross entropy loss. This loss function is equal to applying softmax function $\left[\sigma(y)_i = \frac{e^{y_i}}{\sum_{k=1}^{K} e^{y_k}}\right]$ (where $y$ is a vector of the inputs to the output layer and $i$ indexes the output units, $i = 1, 2, \cdots\cdots K$) and then computing the cross entropy [24]. This loss function was used due to its improved classification accuracy for multiclass problems. Mathematical form of cross entropy loss is defined by Eq. (2).

$$H_{y'}(y) = -\sum_i y_i' \log(y_i) \tag{2}$$

where $y_i$ represents the predicted labels and $y_i'$ represent the true labels.

In addition to these layers, the network is also regularized with drop out layers, having training probability 0.5, to avoid overfitting [29]. The complete architecture of the network is shown in Fig. 3.

### 3.3. Convolutional neural network architecture

The convolutional neural network adopted in this study was also designed using Tensorflow (*Google open source software for Deep Learning*). It had an input layer, 2 convolutional layers, a max pooling layer, a fully connected layer and an output layer. The number of nodes in the input layers were equal to sampling points of signals i.e. 2048. Filter size in the first layer was kept large ($16 \times 1$) because of its good performance in noisy conditions [20]. In order to keep the network simple, there was no pooling layer between the first and second convolution layer and its effect was approximated with a larger stride ($2 \times 1$) in the second convolutional layer as it does not reduce performance, however, reduces the number of layers [25]. After 2nd convolution layer, one max pooling layer was applied with filter size $2 \times 1$ and stride size $2 \times 1$. Proceeding max pool are fully connected layer with 300 nodes and an output layer with 5 nodes. The number of nodes in the fully connected layer were also decided after several trails and chose the number the nodes that gave good performance. The activation function used in convolution layer was the exponential linear unit (Elu) $ELU_g(h) = \begin{cases} h & h \geq 0 \\ g(\exp(h) - 1) & h < 0 \end{cases}$, (*where* $g = -1$) in order to avoid vanishing gradient problems [30] while $Relu(x) = \max(0, x)$ is used in fully connected layer. Beside these, drop out regularization was also employed before and after fully connected layer in order to avoid over fitting. The complete architecture is present in Fig. 4 and the parameters are given in Table 5.

## 4. Performance evaluation

### 4.1. Performance of fully connected DNN on augmented database

Performance of DNN was evaluated for augmented database by training the network on the training database and then testing its performance for testing database. Learning curves for 500 epochs in Fig. 5 shows that DNN started with approximately 40% of testing accuracy, reached the saturation point at approximately 125th epoch and did not improve its performance for another 375 epochs. So, the process was terminated at 500th epoch due to lack of improved results. It can also be seen from learning curves that learning rate is faster for databases with lower noise than for databases for high noise. This swiftness in the learning process can be attributed due to the less complex nature of the signals. The mean of average accuracy achieved in 20 trails with fully connected DNN was 81.51% ± 1.48%, 84.35% ± 1.30%, 89.19 ± 1.25%, 93.40% ± 1.00%, 95.84% ± 0.95% for SNR 5, SNR 7, SNR 10, SNR 15 and SNR 20 respectively as shown in Table 6. It is also quite obvious in learning curves that testing performance keeps on fluctuating around convergence point. This fluctuation is due to oscillation of gradient around global minimum loss. This effect can be minimized by reducing the network learning rate but the tradeoff is much longer training time and gradient entrapment in the local minima. The results show that DNN performs considerably well for low noise signals achieving 95.84% ± 0.95% of accuracy for SNR 20, however, for complex signals having high noise, there is still need for improvements as it achieved only 81.51% ± 1.48% accuracy.

### 4.2. Performance of CNN on augmented database

Performance of CNN was then evaluated on augmented database. The results show that CNN gives significantly better results even for high SNR levels. Learning curves for 500 epochs in Fig. 6 show that CNN starts with approximately 40% of testing accuracy and reach the saturation point at approximately 75th epoch. Here again, the learning rate is faster for signals having low noise while slower for high noise signals. The mean of average accuracy achieved in 20 trails with CNN was 88.33% ± 0.82%, 90.70% ± 0.74%, 93.39% ± 0.72%, 97.73% ± 0.47%, 99.20% ± 0.57% for SNR 5, SNR 7, SNR 10, SNR 15 and SNR 20 respectively as shown in Table 6. The results show that CNN performs considerably well in comparison to DNN even for high noise signals and for low noise, especially for SNR 20, its prediction ability is almost flawless.

### 4.3. Performance comparison

The comparison of DNN and CNN on augmented database shows that CNN gives higher performance than DNN. At SNR 5 CNN on average gives 6.82% better performance while 6.35%, 4.20%, 4.33% and 3.36% improvement in performance for SNR 7, SNR 10, SNR 15 and SNR 20 respectively. It can also be seen that prediction gap between DNN and CNN is higher at high noise levels while for low noise levels there is not much performance gap. This shows that CNN has better adaptability for noise than DNN and proves that CNN is more robust for noisy signals. These results can be seen in Table 6 and Fig. 7.

To compare the improvement in performance achieved by data augmentation against original database, the performance of DNN and

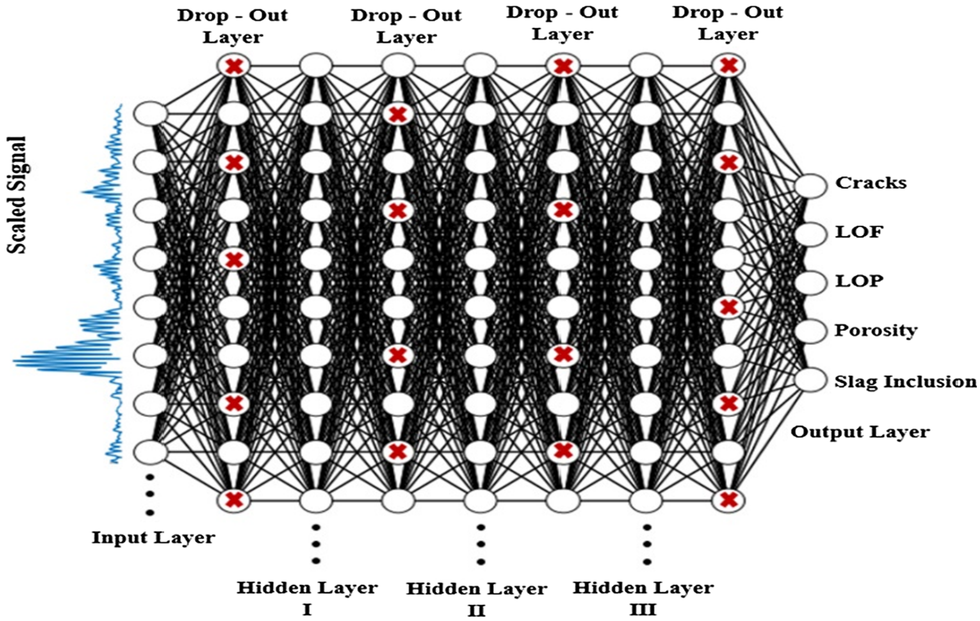


**Fig. 3.** Architecture of fully connected deep neural network.

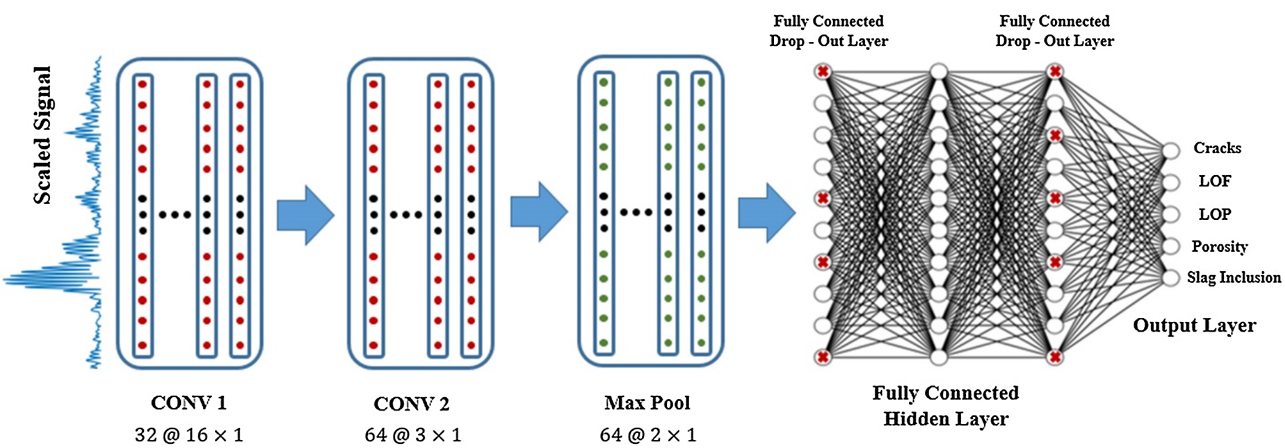


**Fig. 4.** Architecture of convolutional neural network.

**Table 5**
Parameters of convolutional neural network.

| Adopted CNN Architecture | | | | | |
|---|---|---|---|---|---|
| | Layer Type | Kernel Size/ Stride | Features Maps | Output Size | Padding |
| 1 | Conv 1 | 16 × 1/8 × 1 | 32 | 256 × 32 | *Same* |
| 2 | Conv 2 | 3 × 1/2 × 1 | 64 | 128 × 64 | *Same* |
| 3 | Max Pool | 2 × 1/2 × 1 | – | 64 × 64 | *Valid* |
| 4 | Drop Out | 0.25 | – | – | – |
| 5 | Fully Connected | 300 | – | – | – |
| 6 | Drop Out | 0.5 | – | – | – |
| 7 | Softmax with Cross Entropy | 5 | – | 5 | – |

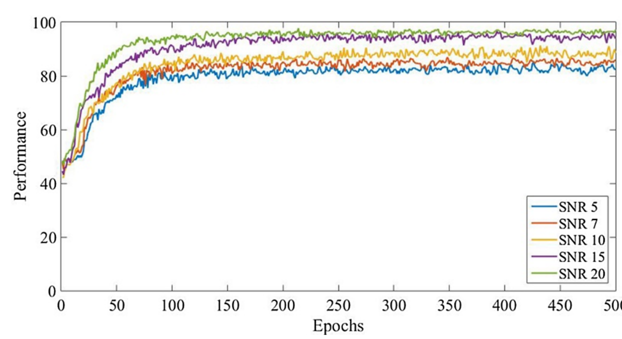


**Fig. 5.** Learning curves of DNN at various SNR levels.

CNN was also evaluated on original database of 720 signals. The original database was divided into training dataset of 648 (90% for each defect type) and testing dataset of 72 signals (10% for each defect type). The performance is then evaluated at each noise level. The results in Table 6 and Fig. 7 shows that augmented database gave considerably improved results for DNN and CNN. For DNN, at SNR 5 it gave 15.84% better results while for SNR 7, SNR 10, SNR 15 and SNR 20, it achieves 13.80%, 14.91%, 13.70% and 13.34% improved performance than original database respectively while for CNN, at SNR 5, it gave 14.85% better results while for SNR 7, SNR 10, SNR 15 and SNR 20, it achieves 12.44%, 11.60%, 13.06% and 12.37% improved performance than original database respectively. This drop in performance for original database having noise can be attributed to the complexity of the signals and very less number of signals in the database. As the signals were complex and very few instances were available for training the neural

**Table 6**
Average performance accuracies of neural networks on original and augmented databases.

| Noise Levels | Original Database | | Augmented Database | |
|---|---|---|---|---|
| | DNN | CNN | DNN | CNN |
| SNR 5 | 65.67% ± 1.68% | 73.48% ± 1.13% | 81.51% ± 1.48% | 88.33% ± 0.82% |
| SNR 7 | 70.55% ± 1.93% | 78.26% ± 1.03% | 84.35% ± 1.30% | 90.70% ± 0.74% |
| SNR 10 | 74.28% ± 1.65% | 81.79% ± 1.21% | 89.19% ± 1.25% | 93.39% ± 0.72% |
| SNR 15 | 79.70% ± 1.50% | 84.67% ± 1.05% | 93.40% ± 1.00% | 97.73% ± 0.47% |
| SNR 20 | 82.50% ± 1.26% | 86.83% ± 0.71% | 95.84% ± 0.95% | 99.20% ± 0.57% |

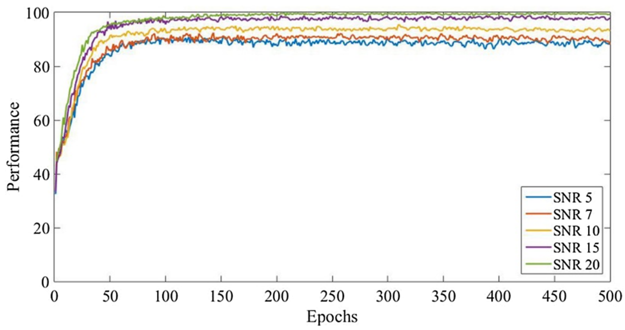


**Fig. 6.** Learning curves of CNN at various SNR levels.

networks, so it overall hampered the learning and classification ability of DNN and CNN. Although, the performance of both DNN and CNN is lower for original dataset than augmented data set, however, CNN still gave better performance than DNN. Comparing their accuracies for original database CNN still gave 7.81%, 7.71%, 7.51%, 4.97% and 4.33% better accuracies for SNR 5, SNR7, SNR 10, SNR 15 and SNR 20 respectively. Here, again, prediction gap between DNN and CNN is higher at high noise levels while for low noise levels, this gap shrinks.

### 4.4. Defect classification

Comparison of DNN and CNN for defect classification accuracy on augmented database was made by generating the confusion matrices at all selected noise levels as shown in Table 7. Results show that DNN was able to easily classify LOF and porosity signals, but for cracks, LOP and especially for slag signals its classification capability reduced for high noise signals. At SNR 5, DNN classified 81.63% ± 1.32% cracks signals correctly, however wrongly predicted 12.03% crack signals as LOF, 2.63% crack signals as LOP, 1.93% crack signals as porosity and 1.78% crack signals as slag inclusions. Considering LOF defect, it classified 87.93% ± 1.85% correctly, but misclassified 9.27% LOF signals as cracks, 0.35% signals as LOP, 0.36% as porosity and 2.09% as slag inclusions.

For LOP defect, it classified 72.14% ± 3.04% signals correctly, but wrongly predicted 27.86% signals. In the case of porosity defect, DNN gave very high accuracy of 90.40% ± 2.35% but made few mistake by classifying 5.97% and 2.26% porosity signal as slag inclusion and cracks respectively. For slag inclusion, DNN gave the worst performance, classifying only 68.13% ± 1.70% slag inclusion signals correctly and misclassified 31.87% of slag inclusion signals as cracks, LOF, LOP and porosity signals. Classification accuracies were 83.44% ± 1.26%, 89.14% ± 1.45%, 83.00% ± 2.19%, 93.71% ± 1.89% and 71.22% ± 1.87% at SNR 7, 87.25% ± 1.12%, 91.80% ± 1.34%, 91.66% ± 1.46%, 96.86% ± 1.37%, 81.55% ± 1.86% at SNR 10, 92.09% ± 1.07%, 92.92% ± 1.08, 96.09% ± 1.26%, 99.80% ± 0.40% and 91.49% ± 0.90% at SNR 15 and 94.61% ± 0.60%, 94.21 ± 1.06%, 98.40% ± 0.89%, 99.89% ± 0.31% and 97.76% ± 1.06% at SNR 20 for cracks, LOF, LOP, porosity and slag inclusion defects respectively as shown in Table 7.

In comparison, CNN defect classification accuracy is significantly better than DNN even at high noise levels. CNN was also a bit confused between cracks, LOF, LOP and slag signals for noisy signals, but the extent of the wrong prediction is considerably lower than DNN. CNN gave 4.59%, 10.80%, 15.45% and 5.73%, 10.29%, 15.87% better accuracy for cracks, LOP and slag inclusion signals at SNR 5 and SNR 7 respectively. CNN also gives better accuracies up to 5.18%, 10.25% at SNR 10 for cracks and slag inclusions signals respectively. At SNR 15 although both performs fairly well, CNN gave 3.96%, 4.88% and 7.98% better accuracies for cracks, LOF and slag signals while at SNR 20 CNN was almost flawless and give 3.15%, 5.62% more accuracy for cracks and LOF than DNN as shown in Table 7 and Fig. 8.

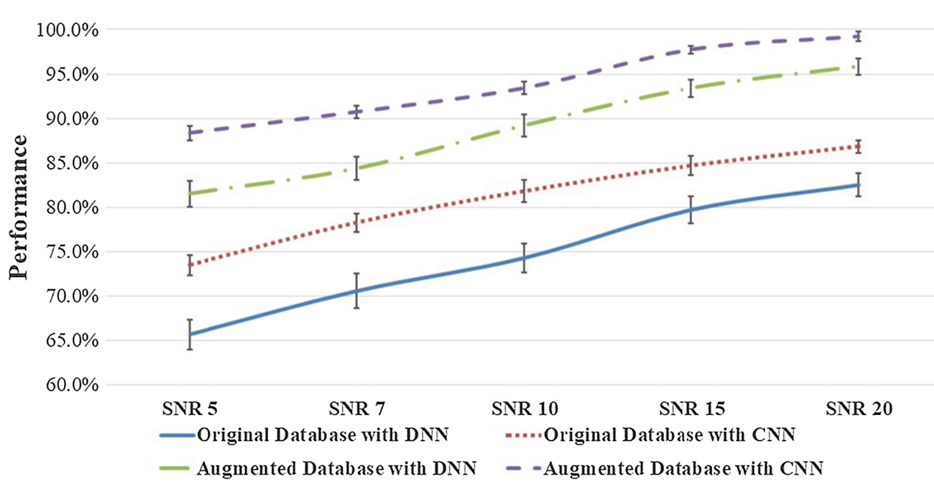


**Fig. 7.** Performance comparison of neural networks on original and augmented databases.

**Table 7**
Confusion matrices (in percentage) of DNN and CNN for augmented database at different noise levels.

| | Fully Connected DNN | | | | | CNN | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **SNR 5** | | | | | | | | | |
| | Cracks | LOF | LOP | Porosity | Slag | Cracks | LOF | LOP | Porosity | Slag |
| **Cracks** | 81.63 ± 1.32 | 12.03 | 2.63 | 1.93 | 1.78 | 86.22 ± 0.84 | 8.79 | 3.85 | 0.70 | 0.44 |
| **LOF** | 9.27 | 87.93 ± 1.85 | 0.35 | 0.36 | 2.09 | 7.22 | 91.83 ± 1.23 | 0.12 | 0.00 | 0.83 |
| **LOP** | 13.57 | 8.77 | 72.14 ± 3.04 | 2.41 | 3.11 | 7.51 | 5.89 | 82.94 ± 2.09 | 0.86 | 2.80 |
| **Porosity** | 2.26 | 0.57 | 0.80 | 90.40 ± 2.35 | 5.97 | 0.49 | 0.00 | 0.00 | 96.91 ± 1.23 | 2.60 |
| **Slag** | 9.69 | 13.07 | 2.07 | 7.04 | 68.13 ± 1.70 | 9.58 | 5.87 | 0.00 | 0.97 | 83.58 ± 0.80 |
| | **SNR 7** | | | | | | | | | |
| **Cracks** | 83.44 ± 1.26 | 10.92 | 2.36 | 1.58 | 1.70 | 89.17 ± 0.68 | 6.53 | 3.50 | 0.17 | 0.63 |
| **LOF** | 9.32 | 89.14 ± 1.45 | 0.19 | 0.16 | 1.19 | 8.37 | 90.87 ± 1.21 | 0.11 | 0.00 | 0.65 |
| **LOP** | 10.23 | 4.34 | 83.00 ± 2.19 | 0.19 | 2.24 | 3.97 | 0.23 | 93.29 ± 1.16 | 0.00 | 2.51 |
| **Porosity** | 1.60 | 0.42 | 0.41 | 93.71 ± 1.89 | 3.86 | 0.17 | 0.00 | 0.00 | 98.43 ± 1.00 | 1.40 |
| **Slag** | 8.15 | 12.78 | 1.25 | 6.60 | 71.22 ± 1.87 | 7.11 | 5.45 | 0.00 | 0.35 | 87.09 ± 0.85 |
| | **SNR 10** | | | | | | | | | |
| **Cracks** | 87.25 ± 1.12 | 9.83 | 1.00 | 1.25 | 0.67 | 92.43 ± 0.99 | 6.65 | 0.03 | 0.10 | 0.79 |
| **LOF** | 7.03 | 91.80 ± 1.34 | 0.44 | 0.00 | 0.73 | 6.88 | 93.10 ± 0.68 | 0.01 | 0.00 | 0.01 |
| **LOP** | 4.29 | 1.35 | 91.66 ± 1.46 | 0.00 | 2.70 | 1.97 | 4.43 | 93.60 ± 1.20 | 0.00 | 0.00 |
| **Porosity** | 1.22 | 0.00 | 0.21 | 96.86 ± 1.37 | 1.71 | 0.03 | 0.00 | 0.03 | 99.91 ± 0.29 | 0.03 |
| **Slag** | 4.75 | 9.38 | 0.01 | 4.31 | 81.55 ± 1.86 | 0.60 | 7.51 | 0.03 | 0.06 | 91.80 ± 0.70 |
| | **SNR 15** | | | | | | | | | |
| **Cracks** | 92.09 ± 1.07 | 6.22 | 0.93 | 0.30 | 0.46 | 96.05 ± 0.51 | 3.92 | 0.03 | 0.00 | 0.00 |
| **LOF** | 6.21 | 92.92 ± 1.08 | 0.46 | 0.00 | 0.41 | 2.18 | 97.80 ± 0.50 | 0.01 | 0.00 | 0.01 |
| **LOP** | 2.66 | 0.00 | 96.09 ± 1.26 | 0.25 | 1.00 | 0.03 | 0.71 | 99.26 ± 0.64 | 0.00 | 0.00 |
| **Porosity** | 0.03 | 0.00 | 0.00 | 99.80 ± 0.40 | 0.17 | 1.00 | 0.00 | 0.00 | 99.00 ± 0.68 | 0.00 |
| **Slag** | 1.38 | 4.15 | 0.33 | 2.65 | 91.49 ± 0.90 | 0.53 | 0.00 | 0.00 | 0.00 | 99.47 ± 0.43 |
| | **SNR 20** | | | | | | | | | |
| **Cracks** | 94.61 ± 0.60 | 5.02 | 0.28 | 0.06 | 0.03 | 97.76 ± 0.65 | 2.24 | 0.00 | 0.00 | 0.00 |
| **LOF** | 4.63 | 94.21 ± 1.06 | 0.26 | 0.00 | 0.90 | 0.17 | 99.83 ± 0.27 | 0.00 | 0.00 | 0.00 |
| **LOP** | 1.37 | 0.00 | 98.40 ± 0.89 | 0.00 | 0.23 | 0.00 | 0.00 | 100.00 ± 0.00 | 0.00 | 0.00 |
| **Porosity** | 0.00 | 0.00 | 0.00 | 99.89 ± 0.31 | 0.11 | 0.00 | 0.00 | 0.00 | 100.00 ± 0.00 | 0.00 |
| **Slag** | 0.53 | 0.47 | 0.52 | 0.72 | 97.76 ± 1.06 | 0.00 | 0.00 | 0.00 | 0.00 | 100.00 ± 0.00 |

## 5. Summary

In this research, an effort was made to more effectively automate ultrasonic defect classification. The focus is to find a robust network architecture that remains independent of feature extraction techniques and give satisfactory performance for noisy signals.

Here, first, the augmented database was synthesized from original database by time shifting the defect signals, as very few signals were available for defect classification in original database. This time shifting technique is analogous to changing the distance between the defect and the transducer. Performance of DNN was then evaluated on this augmented database. Results shows that DNN gave better classification performance for low noise signals but for noisy signals, there is a significant drop in its prediction ability. In order to improve the performance for automatic flaw classification, CNN was then employed, which in contrast to DNN has convolutional layers except fully connected layers and gave 6.82%, 6.35%, 4.20%, 4.33% and 3.36% better performance than DNN at SNR 5, SNR 7, SNR 10, SNR 15 and SNR 20 respectively. The performance of DNN and CNN was also evaluated on original database to show the improvement in their classification ability after data augmentation.

In terms of defect classification accuracy, DNN gave poor performance to classify cracks, LOP and especially slag inclusion defects at high noise levels. However, CNN gave significantly better performance even for high noise signals than DNN and gave 4.59%, 10.80% and 15.45% better performance for cracks, LOP and slag defects at SNR 5. This trend of better performance remains persistent at SNR 7, SNR 10, SNR 15 and SNR 20. At the low noise, especially at SNR 15 and SNR 20 the performance between DNN and CNN was not too different, but CNN still gave better results than DNN for certain defects.

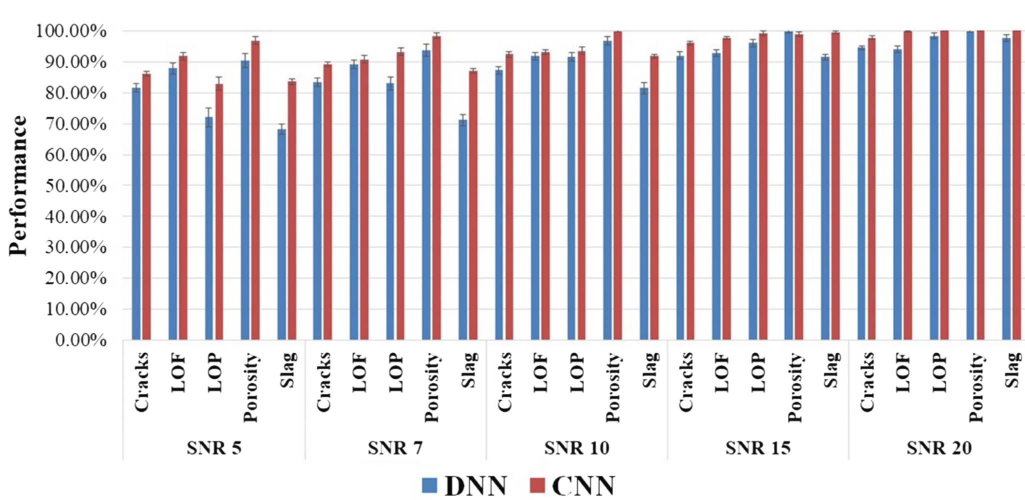


**Fig. 8.** Defect classification comparison between DNN and CNN for augmented database at different noise levels.

Thus, it can be concluded that CNN is the most robust architecture for ultrasonic defect classification for noisy signals and does not require feature extraction procedure for higher performance.

## Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) under grant (No. NRF – 2016R1A6A3A11932440) funded by the South Korean Government (MEST).

## References

[1] S.J. Song, L.W. Schmerr, Ultrasonic flaw classification in weldments using probabilistic neural networks, J. Nondestr. Eval. 11 (2) (1992) 69–77.
[2] A. Masnata, M. Sunseri, Neural network classification of flaws detected by ultrasonic means, NDT and E Int. 29 (2) (1996) 87–93.
[3] F.W. Margrave, K. Rigas, D.A. Bradley, P. Barrowcliffe, The use of neural networks in ultrasonic flaw detection, Measurement 25 (2) (1999) 143–154.
[4] S.W. Liu, J.H. Huang, J.C. Sung, C.C. Lee, Detection of cracks using neural networks and computational mechanics, Comput. Methods Appl. Mech. Eng. 191 (25–26) (2002) 2831–2845.
[5] R. Drai, M. Khelil, A. Benchaala, Time frequency and wavelet transform applied to selected problems in ultrasonics, NDE. NDT & e Int. 35 (8) (2002) 567–572.
[6] S.J. Song, H.J. Kim, H. Cho, Development of an intelligent system for ultrasonic flaw classification in weldments, Nucl. Eng. Des. 212 (1) (2002) 307–320.
[7] F. Cau, A. Fanni, A. Montisci, P. Testoni, M. Usai, Artificial neural networks for non-destructive evaluation with ultrasonic waves in not accessible pipes, Industry Applications Conference, 2005. Fourtieth IAS Annual Meeting, (2005).
[8] S. Sambath, P. Nagaraj, N. Selvakumar, Automatic defect classification in ultrasonic NDT using artificial intelligence, J. Nondestr. Eval. 30 (1) (2011) 20–28.
[9] E.F. Simas Filho, M.M. Silva, P.C. Farias, M.C. Albuquerque, I.C. Silva, C.T. Farias, Flexible decision support system for ultrasound evaluation of fiber–metal laminates implemented in a DSP, NDT and E Int. 79 (2016) 38–45.
[10] F.C. Cruz, E.F. Simas Filho, M.C. Albuquerque, I.C. SIlva, C.T. Farias, L.L. Gouvêa, Efficient feature selection for neural network based detection of flaws in steel welded joints using ultrasound testing, Ultrasonics 73 (2017) 1–8.
[11] N. Munir, H.J. Kim, S.J. Song, S.S. Kang, Investigation of deep neural network with drop out for ultrasonic flaw classification in weldments, J. Mech. Sci. Technol. 32 (7) (2018) 3073–3080.
[12] M. Lin, Q. Chen, S. Yan, Network in network, arXiv preprint, arXiv:1312.4400, 2013.
[13] Z. Gan, R. Henao, D. Carlson, L. Carin, Learning deep sigmoid belief networks with data augmentation, Artif. Intell. Stat. (2015) 268–276.
[14] X. Cui, V. Goel, B. Kingsbury, Data augmentation for deep neural network acoustic modeling, IEEE/ACM Trans. Audio Speech Lang. Process. 23 (9) (2015) 1469–1477.
[15] L. Perez, J. Wang, The Effectiveness of Data Augmentation in Image Classification using Deep Learning, arXiv preprint, arXiv:1712.04621, 2017.
[16] Y. Lecun, L. Bottou, Y. Benjio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[17] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[18] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Networks 61 (2015) 85–117.
[19] M. Meng, Y.J. Chua, E. Wouterson, C.P.K. Ong, Ultrasonic signal classification and imaging system for composite materials via deep convolutional neural networks, Neurocomputing 257 (2017) 128–135.
[20] W. Zhang, C. Li, G. Peng, Y. Chen, Z. Zhang, A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load, Mech. Syst. Sig. Process. 100 (2018) 439–453.
[21] B. Rimoldi, Principles of Digital Communication: A Top-Down Approach, Cambridge University Press, 2016.
[22] R.G. Gallager, Principles of Digital Communication, Cambridge University Press, Cambridge, UK, 2008.
[23] T. Rashid, Make your own neural network, CreateSpace Independent Publishing Platform, 2016.
[24] A. Géron, Hands on Machine Learning with Scikit-Learn and Tensorflow, O'Reilly Media, Inc., 2017.
[25] J. T. Springenberg, A. Dosovitskiy, T. Brox, M. Riedmiller, Striving for simplicity: the all convolutional net, arXiv preprint, arXiv:1412.6806, 2014.
[26] V. Nair, G.E. Hinton, Rectified linear units improve restricted boltzmann machines, Proceedings of the 27th International Conference on Machine Learning (ICML-10), (2010).
[27] A.L. Maas, A.Y. Hannun, A.Y. Ng., Rectifier nonlinearities improve neural network acoustic models, in: Proc. ICML, 2013, June.
[28] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inform. Process. Syst. (2012).
[29] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[30] D.A. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (elus), arXiv preprint, arXiv:1511.07289, 2015.