Gramophone Noise Detection and Reconstruction Using Time Delay Artificial Neural Networks

Christoph F. Stallmann, Andries P. Engelbrecht, *Senior Member, IEEE*

***Abstract*—Gramophone records were the main recording medium for more than seven decades and regained widespread popularity over the past several years. Being an analogue storage medium, gramophone records are subject to distortions caused by scratches, dust particles, degradation and other means of improper handling. The observed noise often leads to an unpleasant listening experience and requires a filtering process to remove the unwanted disruptions and improve the audio quality. This article proposes a novel approach that employs various feed forward time delay artificial neural networks to detect and reconstruct noise in musical sound waves. A set of 800 songs from eight different genres were used to validate the performance of the neural networks. The performance was analysed according to the outlier detection and interpolation accuracy, the computational time and the tradeoff between the accuracy and time. The empirical results of both the detection and reconstruction neural networks were compared to a number of other algorithms, including various statistical measurements, duplication approaches, trigonometric processes, polynomials and time series models. It was found that the neural networks' outlier detection accuracy was slightly lower than some of the other noise identification algorithms, but achieved a more efficient tradeoff by detecting most of the noise in real time. The reconstruction process favoured neural networks with an increase in the interpolation accuracy compared to other widely used time series models. It was also found that certain genres such as classical, country and jazz music were interpolated more accurately. Volatile signals, such as electronic, metal and pop music were more challenging to reconstruct and were substantially better interpolated using neural networks than the other examined algorithms.**

***Index Terms*—gramophone records, noise detection, noise reconstruction, feed forward neural networks, recurrent neural networks, time delay neural networks, audio signal modelling**

I. Introduction

GRAMOPHONE records were first introduced as commercial audio storage medium by Berliner in 1889 [1] based on the research and experimentation of booksellers, poets and scientists such as Scott, Cros, Edison and Bell [2]. Gramophone records were widely used for more than seven decades until they were replaced as main recoding medium by the compact disc (CD) in the late 1980s. Although numerous new audio technologies were introduced during the past couple of decades, gramophones regained widespread popularity over the past few years. Approximately six million records were sold in 2013 in the United States alone, a 33% increase from the previous year [3]. However, it was revealed that these figures are an underestimate and only represent 15% of the total sales, since bar codes, which are used to track sales, are absent from most records [4].

Christoph F. Stallmann and Andries P. Engelbrecht are with the Department of Computer Science, University of Pretoria, Pretoria, South Africa (email: cstallmann@cs.up.ac.za, engel@cs.up.ac.za).

The first gramophones originated half a century before the first digital computer and master recordings prior to the 1960s are, therefore, only available on gramophone, making them invaluable to historic archives. Museums, commercial music labels and private audiophiles are still digitizing these rare records to date, a process that is tedious and time consuming, since most recordings are damaged and need to be refurbished and remastered. Since gramophone records are an analogue medium, they are subject to distortions which are mainly caused by physical scratches. Excessive playback, increased temperatures, dust particles and imperfections in both the record and the turntable mechanics also cause minor noise in the reproduced signal.

This article presents a novel approach that detects and reconstructs noise in gramophone audio signals by employing various feed forward artificial neural networks (ANN) to predict and interpolate the distorted sound waves. The performance of the proposed system is compared to other widely used noise detection and interpolation algorithms which are based on numerous statistical measurements, trigonometric and duplication approaches, polynomials and time series models. Section II briefly discusses the problem at hand with the current state of the art of ANNs in audio processing given in section III. Section IV presents the time delay ANN (TDANN) for predictive noise detection, followed by the TDANN noise reconstruction process in section V. The random initial weights of the networks are discussed in section VI. The test dataset, performance measurement, computational time and the tradeoff between the detection and reconstruction accuracies and the execution time is given in section VII. The examination of the optimal parameters, including the activation function, training algorithm and epochs, learning rate and momentum, and the network structure, follows in section VIII. The different TDANNs are benchmarked in section IX and compared to the performance of the polynomials and time series models. A detailed comparison between the investigated TDANNs and other noise detection and reconstruction algorithms is given in the appendices.

II. Problem Statement

Gramophone records are subject to distortions, with the majority of the audible noise caused by physical damage to the record. Scratches cause the stylus' needle to move in an undesirable direction, resulting in disruptions in the

reproduced sound wave. These disruptions are in most cases easily identified by the human ear and are commonly referred to as *crackles*, *pops* or *clicks*. A few examples of noise caused by scratches are given in figure 1. Although other kinds of noise, such as dust in the grooves, and turntable rumble and vibrations, also influence the reproduced signal, the affected frequencies are typically not observable by the human ear and are therefore not considered further in this study and left for future research. In order to improve the sound quality, the noise

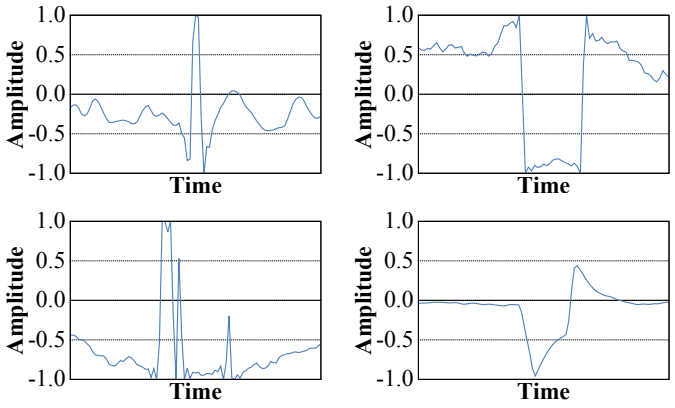

Fig. 1. Typical disruptions caused by scratches on gramophone records.

has to be removed from the signal. Two main approaches exist in audio processing for noise removal. The first approach is typically easier to implement and filters the entire signal, either in the time or the frequency domain, adjusting all samples in the signal [5]. Hence, not only are the noisy samples adjusted in order to remove the distortions, but also the segments of the signal that are not affected by noise, causing undesirable information loss. The second approach is computationally more expensive, since two distinct steps have to be conducted for the noise removal. The first phase detects all the noisy samples which are then corrected in the second step, leaving the remainder of the non-noisy signal untouched [6], [7].

This article uses the latter approach which detects the noise before reconstructing the disruptions. Since only the noisy samples are adjusted, compared to all samples in the former approach, the final output signal is of higher quality, providing a more accurate estimation of the original signal. Both the detection and reconstruction processes are addressed using TDANNs, since audio data are time delayed signals and therefore benefit from the properties of time delayed models [8], [9].

III. State of the Art

ANNs are well established in audio processing, with the majority of current research focusing on speech processing, such as speech recognition [10], speech enhancement [11], and speech tagging [12]. Other areas of audio processing also benefit from ANNs, such as audio classification [13], feature extraction [14], and audio declipping [15]. However, comparatively little research has been done on the detection and removal of impulse disturbances from musical signals. The signals are recorded from older technologies, such as gramophone records or magnetic tapes, compared to modern digital music which is not affected by these disruptions.

Czyzewski proposed an ANN to detect impulse disturbances which are then reconstructed with another ANN [16]. The approach only achieved a mean squared error (MSE) of 0.25 requiring 264 minutes of training on only 10 patterns. Czyzewski concluded that the long training time limits ANNs for practical use. In addition, detecting various impulses requires the assembly of a large knowledge base of universal training patterns, which makes the approach ineffective for gramophone refurbishment, since scratches cause a wide range of non-universal disruptions. The research was later extended with ANNs using self organizing maps (SOM) and rough sets, with computational time still being a major issue [17].

Cocchi and Uncini used a forward and backward prediction subband ANN to reconstruct large gaps of 200 to 5000 missing samples [18]. The approach achieved a reconstruction MSE between 0.002 and 0.04, with no reference given to the computational time.

Seng et. al. proposed a radial basis function ANN with forward and backward prediction for the reconstruction of missing samples [19]. The authors did not provide any results, neither indicating the interpolation performance, nor the computational time required.

This article utilizes a number of different ANNs to reconstruct samples in the time domain, in contrast to using subbands in the frequency domain [18], [19]. Various architectures were utilized in order to determine the performance benefits for each architecture, both from an interpolation accuracy and computational time perspective. Seng et. al. [19] did not conduct an empirical analysis on the proposed algorithm, whereas [16]–[18] only ran experiments on a small number of audio segments from isolated songs. This article uses a large set of songs in order to verify that the algorithms perform well on average over the entire dataset and not only on individual audio segments.

IV. Noise Detection

Disruptions caused by scratches on gramophone records produce sudden bursts in the sound wave. The noise or outliers are detected by calculating the deviation of the surrounding samples in the time domain or by determining the degree of anomalies in the frequency domain [20], [21].

An alternative outlier detection approach in time series analysis is to employ a forecasting model to predict the next few samples in a series. If the predicted samples deviate from the observed values with a certain degree, they are marked as outliers. Various predictive outlier detectors were proposed for time series, utilizing models such as multilayer perceptrons [22], autoregressive models [23], [24], support vector regression [25], and nearest cluster prediction [22].

The absolute predictive deviation (APD) using a forecasting model m on a series y is calculated at time delay $t+1$ as follows:

$$apd(y) = |y_{t+1} - m(y_{t-n+1}, \ldots, y_t)| \quad (1)$$

where n is the number of input samples used to approximate m. An alternative approach replaces the absolute Euclidean deviation with the Mahalanobis distance [26]. The APD generates a noise map η containing a value in $[0, s]$ for each sample in the observed signal where s is depended on the output of the model m. A binary noise mask $\breve{\eta}$ is generated by subjecting the noise map to a threshold ϕ as follows:

$$\breve{\eta}_i = \begin{cases} 1 & \text{for } \eta_i \geq \phi \\ 0 & \text{for } \eta_i < \phi \end{cases} \tag{2}$$

where integer i ranges over all samples in the observed signal.

Applying a threshold in this manner is sound for univariate outliers. However, if multivariate outliers are present in a low volatile signal, which is the case with most music signals, the model estimation from time delay $t + 2$ onwards is skewed if the sample at delay $t + 1$ was flagged as an outlier. If the sample y_{t+1} was identified as noise, samples $y_{t+2}, \ldots, y_{t+k}$ in the multivariate outlier stretching over k data points are likely not flagged as noise, since the models from $t + 2$ onwards are approximated using noisy input data. In addition, the inlier samples immediately succeeding the multivariate outliers from time delay $t + k + 1$ are likely to be incorrectly flagged as noise if the preceding outlier substantially deviates from the surrounding signal. In order to circumvent this problem, one of two alternative approaches are utilized.

The first approach, batch prediction, only estimates the model m once for a single multivariate outlier. If y_{t+1} was flagged as noise, all the k subsequent samples are predicted at once and then subjected to the threshold. Batch prediction is computationally efficient, but relies on the model's ability to accurately predict all k samples. In addition, the model requires preknowledge about the maximum possible duration of an outlier. This poses a problem, especially with standard feed forward ANNs which require the number of output neurons to be fixed before training commences. A simple solution is to choose a sensible number of output neurons that will accommodate all noise durations. However, too many unnecessary output neurons increase the training and execution time.

The second approach, recurrent prediction, only forecasts one sample at a time. If y_{t+1} was identified as outlier, its value is replaced before the next model is approximated at delay $t + 2$. The sample is either replaced by the previous prediction, the last known inlier, or a weighted average of the preceding n samples. Recurrent prediction is computationally more expensive than batch prediction, since m has to be estimated k times instead of only once. However, since models typically struggle to predict far into the future, recurrent prediction has an advantage over batch prediction by only forecasting a single sample at a time.

This article investigates two alternative feed forward TDANNs [27] that are employed for the APD outlier detection algorithm. On initialization, the TDANNs have a value for only the first input neuron, all other inputs are set to zero. Once the first pattern was propagated through the TDANN and a second pattern is available, all input values are shifted ahead by one, therefore, introducing a time delay t. The two alternative TDANNs are implemented as follows:

- Predictive TDANN (APD-TDANN): The TDANN is trained incrementally on the input signal. The network has K output neurons where K is the anticipated number of samples of the largest multivariate outlier in the signal. K should be chosen cautiously, since a smaller K only detects short multivariate outliers no longer than K samples. On the other hand, a larger K is able to detect longer noise durations, but has a substantial increase in the training time. Therefore, K should be chosen to accommodate the largest possible multivariate outlier, but at the same time should be small enough in order to not unnecessarily increase the computational time. The structure of the network is illustrated in figure 11.
- Predictive simple recurrent TDANN (APD-SRTDANN): A novel approach in time series analysis is to combine a TDANN and a simple recurrent ANN (SRTDANN) in order to learn the temporal characteristics of the latests samples of the series [28]–[30]. A Jordan simple recurrent ANN [31] is trained incrementally, containing a single output neuron linked with a recurrent connection as an additional neuron to the input layer. Unlike the APD-TDANN, this architecture does not require any preknowledge about the noise duration and can therefore predict any multivariate outlier duration without adjusting the output layer. In addition, the training for this architecture is faster than the APD-TDANN, since the output layer only contains a single neuron. The network structure is given in figure 13.

V. Noise Reconstruction

Once the noise detection process concludes, the input signal y and the binary noise mask $\breve{\eta}$ are presented to the reconstruction algorithm. The reconstruction algorithm treats multivariate outliers as a gap of missing samples and generates a new signal z by replacing each flagged outlier in y with the output values of model m as follows:

$$z_i = \begin{cases} m_i(y) & \text{for } \breve{\eta}_i = 1 \\ y_i & \text{for } \breve{\eta}_i = 0 \end{cases} \tag{3}$$

This article analyses various TDANNs that are utilized to interpolate the gaps of previously identified noisy samples. Given K as the sample length of the largest multivariate outlier in the signal and k as the length of the current gap to be interpolated for $1 \leq k \leq K$, the reconstruction process applies the following TDANN architectures:

- Forward incremental TDANN (FI-TDANN): A TDANN is trained by shifting a moving window over the input signal and sequentially presenting the samples as inputs to the network. Incremental training updates the weights using only the samples preceding the gap. The TDANN has K output neurons. Therefore, if a gap of k samples is encountered, the TDANN predicts a series of K samples and only uses the first k outputs for the interpolation. Figure 11 illustrates the structure and the manner in which consecutive samples are presented to the network.

- Bidirectional incremental TDANN (BI-TDANN): The previously discussed FI-TDANN only operates on historical data without considering the future direction of the signal. Variants of bidirectional ANNs have shown promising results in other areas of audio processing [32]–[34]. This approach incrementally trains two separate BI-TDANNs, one processing the signal from start to end, the other one from end to start. The gap is then reconstructed with the average output of both BI-TDANNs. The forward network is given in figure 11 and figure 12 shows the network processing the signal backwards.
- Forward incremental SRTDANN (FI-SRTDANN): A Jordan SRTDANN is trained incrementally on historic data. The network has only one output neuron which is recurrently linked as an additional neuron in the input layer. The FI-SRTDANN repeatedly predicts one sample at a time until the entire gap is reconstructed. The network structure is illustrated in figure 13.
- Bidirectional incremental SRTDANN (BI-SRTDANN): Similar to the FI-SRTDANN, two distinct Jordan SRTDANNs are trained on the signal in the positive and negative time directions. The average output of both networks are used to interpolate the samples. The forward and backward network structures are given in figure 13 and figure 14 respectively.
- Forward separate batch TDANN (FSB-TDANN): A set of training patterns is generated with the historical data of the observed signal. The weights of the network are updated once for all patterns during an epoch by employing a batch training algorithm. A total of K TDANNs are maintained, that is, one TDANN for each possible gap size. Each network has a different number of output neurons, equivalent to the number of samples to be reconstructed. If a gap of k samples is encountered, the corresponding k^{th} network is selected from the set, batch trained on the patterns and then used to reconstruct the k missing samples. The structure of the network is given in figure 15.
- Bidirectional separate batch TDANN (BSB-TDANN): The signal is processed in both directions using a set of FSB-TDANNs and the average of both reconstruction processes is used to interpolate the gap. No additional TDANNs are maintained, since one of the K TDANNs from the forward process is simply reused for the backward interpolation. The structure of the forward network is given in figure 15, whereas the backward network is illustrated in figure 16.
- Forward complete batch TDANN (FCB-TDANN): Training patterns are generated using the historical samples of the signal. The network has K output neurons and the weights are updated using a batch training algorithm. If a gap of k samples is interpolated, only the first k outputs from the K output neurons are used. Figure 11 shows the structure of the network.
- Bidirectional complete batch TDANN (BCB-TDANN): Two separate FCB-TDANNs are trained, one for the forward interpolation and one for the backward interpolation of the signal. The average output of both FCB-TDANNs are used to reconstruct the samples. Figure 11 and figure 12 show the forward and backward structures respectively.
- Interpolation batch TDANN (IB-TDANN): The TDANN has two sets of inputs which are delayed by k samples. The first set of inputs consists of the consecutive samples preceding the gap, the second set created from the samples succeeding the gap. A total of K IB-TDANNs are batch trained, one for each possible gap duration. When a gap of size k is reconstructed, the k^{th} network is selected, trained with the patterns and then utilized to interpolate the missing samples. The network structure is given in figure 17.

VI. Initial Weights

The ANNs are initialized with random weights and gradient-based algorithms are used for the training. Since the utilized gradient-based training algorithms are hill climbers, which are notorious for getting stuck in local minima of the error function [35], the initial random weights play an important role in the performance of the ANN. Hence, there is no guarantee for the ANN to ever reach the global minimum. In order to reduce the chances of the ANN to converge to a local minimum, an ensemble ANN may be used or training the ANN using particle swarm optimization (PSO). The ensemble technique trains a set of ANNs in parallel, where each network is initialized with different random weights [36]. In a PSO, each particle represents a candidate ANN where the network weights are used for the particle's vector [37]. However, very good results were achieved with the default initial random weights, indicating that random weights do no pose a major problem in the given situation. Training using PSOs or ensemble ANNs are left for future research.

VII. Empirical Procedure

This section discusses the procedure, test data and noise generation used for the experiments. The performance measurements for the detection and reconstruction algorithms are examined, followed by the execution time and the evaluation of the tradeoff between the performance and the computational time.

A. Test Data and Noise

Empirical tests were conducted on a set of 800 songs divided into eight genres, namely classical, country, electronic, jazz, metal, pop, reggae and rock music, each consisting of 100 tracks. In order to evaluate the performance of the detection algorithms in a controlled environment, the songs were subjected to artificially generated noise. The reconstruction algorithms are not influenced by artificial noise, since outliers are treated as a gap of missing samples. An additional 83 songs were recorded from gramophones where the noise was flagged manually. The 83 tracks served as a validation set to ensure that the artificially generated and real gramophone noise are homogeneous and do not cause a significant deviation in the algorithms' performance.

In audio processing, clean audio data is commonly distorted

using Gaussian white noise [38]–[40]. A more advanced approach suggested to integrate positive pulses with a constant magnitude and a mixture of white noise and impulses [41]. In order to represent gramophone distortions more precisely, the experiments for this article subjected the test data to four different types of artificial generated noise which resemble the bursts caused by scratches on records. The distortions were generated using a combination of both positive and negative pulses which were exposed to a Gaussian white noise process in order to create more unpredictable outliers. Most scratches do not affect more than 30 sequential samples. However, the benchmarking was conducted on multivariate outliers of up to 50 samples to accommodate possible longer disruptions as well. Although the results in this article only represent gaps of up to 50 samples, the algorithms are able to accommodate noise of any duration without changes to their implementations.

TABLE I
THE SAMPLE MEAN, SAMPLE STANDARD DEVIATION AND SAMPLE PEARSON CORRELATION COEFFICIENT OF THE DATASET.

Genre	μ	σ	Positive PCC	Negative PCC
Classical	-0.0005	0.010	0.76 [0.43, 0.91]	-0.73 [-0.36, -0.90]
Country	-0.0008	0.029	0.67 [0.26, 0.86]	-0.68 [-0.28, -0.88]
Electro	+0.0001	0.054	0.49 [0.00, 0.80]	-0.51 [-0.02, -0.80]
Jazz	-0.0002	0.027	0.69 [0.29, 0.88]	-0.65 [-0.22, -0.86]
Metal	-0.0002	0.053	0.54 [0.05, 0.82]	-0.61 [-0.17, -0.85]
Pop	-0.0001	0.046	0.53 [0.04, 0.81]	-0.55 [-0.07, -0.82]
Reggae	-0.0003	0.043	0.56 [0.09, 0.83]	-0.59 [-0.13, -0.84]
Rock	-0.0016	0.032	0.59 [0.13, 0.84]	-0.63 [-0.19, -0.86]
Average	-0.0004	0.037	0.60 [0.15, 0.85]	-0.62 [-0.17, -0.85]

The Pearson correlation coefficient (PCC) is a statistical measurement of the linear dependency between two variables x and y [42], [43]. The coefficient lies in $[-1, 1]$, where zero indicates no linear dependency and a positive or negative one describes a perfect linear correlation. The sample PCC is defined as

$$r = \frac{\sum_{i=1}^{n}(x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_{i=1}^{n}(x_i - \mu_x)^2}\sqrt{\sum_{i=1}^{n}(y_i - \mu_y)^2}} \tag{4}$$

where μ_x and μ_y are the sample means of x and y respectively.

Table I shows the sample mean, sample standard deviation and sample PCC for the test dataset, calculated using a sample size of 32. The sample mean of all genres was close to zero. The standard deviation shows that electronic and metal music had the highest volatility, whereas classical, country and jazz signals were more stable with a strong linear dependency as indicated by a PCC between ± 0.65 and ± 0.76. With an increase in volatility in the electronic and metal genres, the linearity decreased, although the linear dependency was still moderate at approximately ± 0.5 to ± 0.6. The 95% PCC confidence intervals are given in brackets next to the correlation coefficients in the table.

The songs were encoded using the Free Lossless Audio Codec (FLAC) with stereo channels at a sample rate of 44.1 kHz and an integer sample size of 16 bits.

B. Performance Measurement

The noise detectors were evaluated according to their ability to identify outliers and at the same time keep the incorrectly flagged inliers, which are the samples unaffected by noise, to a minimum. The true positives (TP) and true negatives (TN) are the number of correctly identified outliers and inliers respectively, whereas the false positives (FP) and false negatives (FN) are the number of incorrectly flagged inliers and outliers respectively. The sensitivity (SEN) is the algorithm's capacity to which outliers are correctly flagged. The specificity (SPE) on the other hand represents the ability to correctly recognize inliers. The SEN and SPE is calculated as

$$\text{SEN} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad \text{SPE} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{5}$$

The Matthews correlation coefficient (MCC) was employed as a combined measurement to determine the capability of correctly identifying outliers and penalizing mislabelled inliers [44]. The MCC is computed using

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{6}$$

The MCC lies in $[-1, 1]$ where a negative one indicates a total disagreement, zero a random prediction and a positive one a perfect correlation.

The reconstruction performance was measured using the normalized root mean squared error (NRMSE) in $[0, 1]$ between the original signal y and the reconstructed signal $\tilde{y}$. The NRMSE is defined as

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{\text{n}}\sum_{\text{i}=1}^{\text{n}}\left(\tilde{\text{y}}_\text{i} - \text{y}_\text{i}\right)^2}}{\hat{y} - \check{y}} \tag{7}$$

for a sequence of n samples, where a NRMSE of zero indicates a perfect reconstruction. $\hat{y}$ and $\check{y}$ are the maximum and minimum amplitudes of the original signal respectively.

C. Computational Time and Tradeoff with Performance

The algorithms were also compared according to their computational time. The time is measured as the number of seconds required to process a second of audio data from the input signal, denoted as s\s. A score of one s\s or lower indicates that the algorithm can be executed in real time.

Based on the concept of the scoring metric in [45], in order to assess the tradeoff between an algorithm's performance and the required time, the speed-accuracy-tradeoff (SAT) is calculated using

$$\text{SAT} = \left(\frac{\kappa}{\hat{\kappa} - \check{\kappa}} + \frac{\tau}{\hat{\tau} - \check{\tau}}\right)^{-1} \tag{8}$$

where τ is the computational time measured in s\s. The fastest and slowest execution times are represented by $\hat{\tau}$ and $\check{\tau}$ respectively. The upper bound $\hat{\tau}$ is set to one in order to avoid skewed scores by increasing the penalty for algorithms that take considerably longer to execute than the rest. Since a lower NRMSE specifies a better interpolation accuracy, κ is set to the NRMSE for the reconstruction SAT. However,

since a higher MCC indicates a better detection performance, κ is equal to $(1 - \mathrm{MCC})$ for the noise identification SAT. $\hat{\kappa}$ and $\check{\kappa}$ are the accuracies of the best and worst performing algorithms respectively. A higher SAT score indicates a more efficient tradeoff between the accuracy and the execution time.

Benchmarking was conducted using a single thread on an Intel Core i7 2600 at 3.4 GHz machine with 16 GB of memory running a 64 bit Linux Ubuntu distribution.

VIII. Parameter Optimization

The parameters of the ANNs were optimized using fractional factorial design which is a sensible chosen subset of the experimental executions of a full factorial design [46]. A set consisting of ten songs from each genre was benchmarked to find the optimal parameters. The parameter configurations that on average performed best over all 80 songs in the test set were used to calculate the outlier detection and reconstruction accuracy of the entire validation set of 800 songs. The activation function, training algorithm, epochs, training patterns, learning rate, learning momentum and the network structure which were all optimized using fractional factorial design is discussed below, followed by the optimal parameter configuration for each ANN.

A. Activation Function

Audio samples are real numbers in $[-1, 1]$ and symmetric activation functions were therefore utilized for the TDANNs. Various well known activation functions were tested, including the symmetric sigmoid function, better known as the hyperbolic tangent (tanh) function, a bounded linear activation function, the symmetric Gaussian function and the symmetric trigonometric sine and cosine functions [47]. Due to the large number of samples in an audio track, the activation function is evaluated millions of times and efficiency is therefore of utmost importance. The tanh function is defined as

$$f(net - \theta) = \frac{e^{\alpha(net-\theta)} - e^{-\alpha(net-\theta)}}{e^{\alpha(net-\theta)} + e^{-\alpha(net-\theta)}} \tag{9}$$

for a net input net and a threshold θ. The gradient is controlled by the parameter α. The computation of (9) requires the evaluation of at least two exponents and can be more efficiently approximated using

$$f(net - \theta) = \frac{2}{1 + e^{-\alpha(net-\theta)}} - 1 \tag{10}$$

Even with a single exponential operation, the tanh function is computationally expensive. Elliot proposed an activation function that produces similar results to the sigmoid and tanh functions, but is more efficient to calculate due to absence of exponential operations [48]. The Elliot function produces outputs in $(0, 1)$, and is defined as

$$f(net - \theta) = \frac{\frac{net-\theta}{2}}{1 + |net - \theta|} + \frac{1}{2} \tag{11}$$

The symmetric Elliot function in $(-1, 1)$ is even more efficient to calculate with one less division and addition operation, and is defined as

$$f(net - \theta) = \frac{net - \theta}{1 + |net - \theta|} \tag{12}$$

The Elliot functions reach their extremes more slowly than the sigmoid and tanh functions. Therefore, the output error will in general be greater, requiring more training epochs to reach the desire error [47]. Even though the Elliot function is faster to compute than tanh, the additional training increases the required time again.

The output error and computational time using different activation functions is given in table II. The symmetric Elliot function performed best for all ANNs, except for the recurrent ANN, which had a notably lower output error when using the tanh activation function. The Elliot function was also more efficient than the tanh activation function as indicated by the lower execution time for all ANNs. The tanh activation was utilized for the SRTDANNs and the symmetric Elliot function for all other architectures. Although the bounded linear activation function performed well, it was still inferior to the Elliot and tanh functions. Audio samples in music and speech typically have a zero mean and are distributed according to a Gaussian function with an expected value close to zero. Hence, the extremes of the audio signal at -1 and 1 are rarely reached. Therefore, predicting or interpolating music signal amplitudes favours the Elliot and tanh functions which reach their extremes more slowly and in general have a smaller output error when predicting amplitudes close to the extremes. A linear activation function reaches an amplitude of ± 1 notably faster, resulting in a larger output error on average.

B. Training Algorithm

The TDANNs were trained using supervised learning. A widely used training algorithm is backpropagation which utilizes gradient descent to find a local minimum [49], [50]. Backpropagation can be applied in an incremental, stochastic or batch fashion. Incremental or online backpropagation updates the weights after each presentation of a new training pattern and has proven effective for training TDANNs, especially for processing audio and speech due to the slow changes in the signal [51]–[54]. Batch backpropagation updates the weights only once after all training patterns were presented.

Quickprop is a variation of the backpropagation algorithm which is loosely based on Newton's method [55]. Quickprop requires the second order derivatives of the error function and tries to approximate the error surface with a parabola. Quickprop attempts to use a single step to jump from the current gradient position directly into the minimum of the parabola and, therefore, in general trains faster than standard backpropagation.

Resilient backpropagation (Rprop) is another widely used learning algorithm that penalizes previous weight adjustments which were too large, causing the ANN to jump over local minima [56]. If the partial derivative of a weight changes sign compared to the previous iteration, Rprop decreases the weight update by a factor η^- for $\eta^- < 1$, and increases the weight update by a factor η^+ for $\eta^+ > 1$ if the sign of the partial derivative does not change. Riedmiller and Braun suggested η^- to be fixed at 0.5 and η^+ at 1.2 [57]. iRprop is an improved version of Rprop which typically decreases the training time and was shown to outperform the standard Rprop

and quickprop algorithms [58]. Variations of both Rprop and iRprop exist. Weight-backtracking is implemented in Rprop$^+$ and iRprop$^+$, whereas Rprop$^-$ and iRprop$^-$ omit weight-backtracking [59].

Figure 2 illustrates the average output NRMSE of the batch trained ANNs using different training algorithms. Standard batch backpropagation and quickprop performed better for short training durations, but were inferior to the iRprop algorithms when trained over more epochs. iRprop$^+$ and iRprop$^-$ produced similar results and the deviation in their error increased as training continued. iRprop$^-$ was employed for all batch trained ANNs, since it produced the overall lowest output error when trained for a sensible number of epochs.

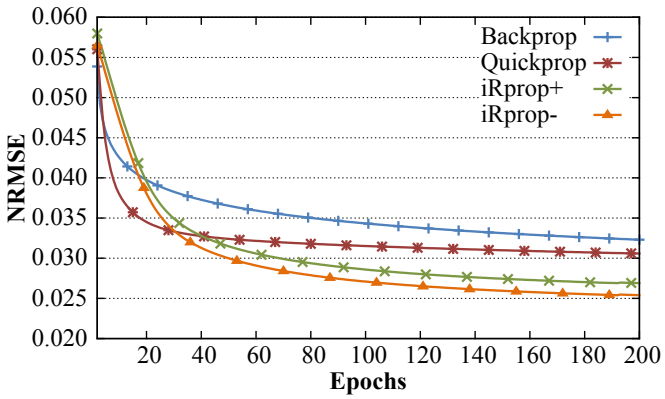

Fig. 2. The average batch training error for different training algorithms.

C. Training Epochs and Patterns

Incremental training for the FI-TDANN, BI-TDANN, FI-SRTDANN and the BI-SRTDANN has an acceptable execution time, since each pattern is only presented once during the updates of the weights. On the other hand, batch training for the FSB-TDANN, BSB-TDANN, FCB-TDANN, BCB-TDANN and the IB-TDANN presents each training pattern p times to the network during the weight updates, where p is the number of training epochs. Given a typical four minute song encoded with stereo channels at a sample rate of 44.1 kHz, the batch trained ANNs are presented with more than 21 million patterns, resulting in an unacceptably slow execution time. In addition, a music signal is relatively stable and changes little over time compared to other more volatile signals. The slow change causes most consecutive training patterns to have very similar characteristics when moving the window one sample at a time over the input signal. In order to reduce the computational time of batch training, the number of training patterns were limited to a maximum u. Training patterns were also accumulated by delaying the consecutive patterns by v samples, compared to the traditional implementation of a TDANN where patterns are delayed by one sample. Both u and v were determined empirically. The lowest value for u and the highest value for v were chosen without increasing the ANN's output error compared to the performance of the ANN when v was set to one and u was equal to the maximum number of patterns preceding the gap. Hence, by employing a pattern count limit u and introducing a pattern delay v, the training time was substantially reduced without affecting the output error of the ANN.

Figure 3 shows the training output error for an increasing number of training epochs. The corresponding interpolation error is illustrated in figure 4. The training error only stabi-

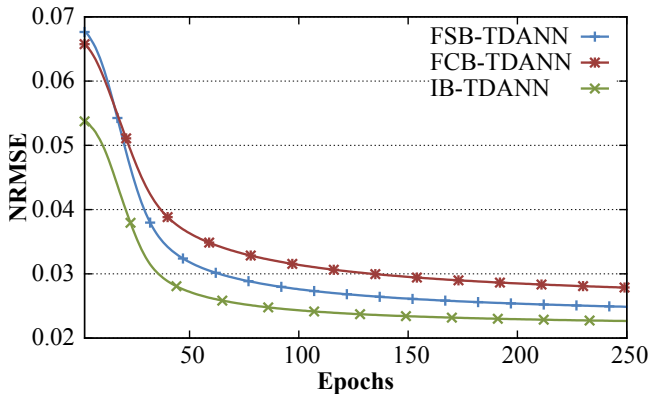

Fig. 3. The training output error for different training durations.

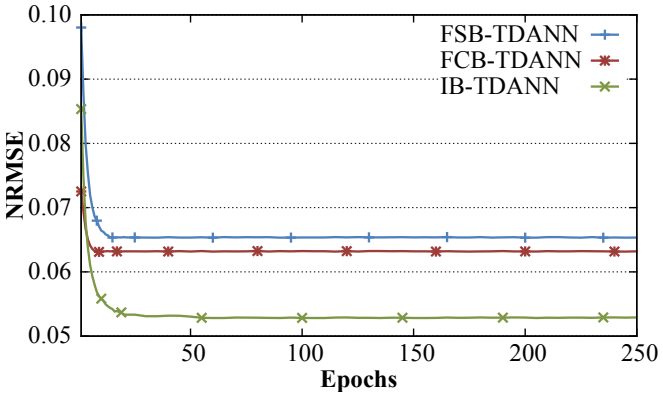

Fig. 4. The interpolation output error for different training durations.

TABLE II
THE INTERPOLATION ERROR AND COMPUTATIONAL TIME FOR DIFFERENT ACTIVATION FUNCTIONS.

Activation Function	FI-TDANN		FI-SRTDANN		FSB-TDANN		FCB-TDANN		IB-TDANN	
	NRMSE	Time	NRMSE	Time	NRMSE	Time	NRMSE	Time	NRMSE	Time
Symmetric Elliot	**0.071308**	64.46232	0.079066	**2.502488**	**0.080021**	52.02565	**0.080626**	**89.62463**	**0.058749**	100.0583
Hyperbolic Tangent	0.075551	64.85891	**0.072680**	2.546073	0.080583	53.08238	0.080856	94.03838	0.058816	101.5186
Bounded Linear	0.079348	64.11473	0.074298	2.540985	0.081192	**51.74124**	0.081005	92.23812	0.058820	**98.67043**
Symmetric Gaussian	0.122094	64.66294	0.106832	2.577210	0.422160	53.25786	0.195167	94.30679	0.341095	101.2180
Symmetric Sine	0.077475	**63.17457**	0.073495	2.515352	0.080711	53.45751	0.080937	94.54052	0.059552	101.9032
Symmetric Cosine	0.105983	64.73592	0.119519	2.532297	0.348429	54.59695	0.122637	95.07163	0.257964	102.9752

lized around the 100^{th} epoch, whereas the interpolation error converged more quickly. The interpolation error of the FSB-TDANN, FCB-TDANN and IB-TDANN steadied after the 16^{th}, 10^{th} and 34^{th} epoch respectively. The maximum number of epochs was set to 50 for all batch trained TDANNs.

D. Learning Rate and Momentum

The influence of the learning rate and momentum on the output error of the incrementally trained ANNs is given in figure 5. Since Rprop is an adaptive training algorithm, the batch trained ANNs do neither require a learning rate nor a learning momentum. The autocovariance of music signals is small, meaning that the signal changes little over time. Incremental training performed better with lower learning rates, where smaller consecutive weight updates corresponded to the gradual transition of the input signal. The learning in incrementally trained TDANNs performed best at a rate of 0.1 and without a momentum.

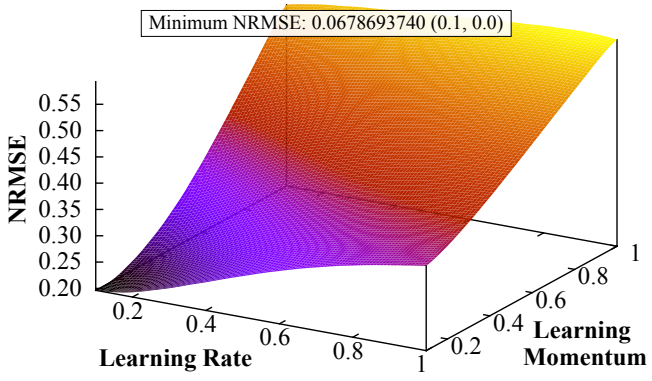

Fig. 5. The output error for different learning rates and momentums.

E. Network Structure

By itself, an artificial neuron is capable of learning linearly separable functions if the summation unit is used [60]. In order to realize complex non-linearly separable functions, neurons are combined into a network architecture. It was proven that a single hidden layer with a sufficient number of neurons can be used to model any continuous function [61]–[63].

The TDANNs' structures were optimized using factorial design in order to determine the ideal number of hidden layers and neurons. An input layer with up to 1536 neurons and three hidden layers with up to 512 neurons each was tested and formed the bounds of the factorial design search. It was found that a set of perceptrons, that is an ANN without any hidden layer, performed best. These findings are supported by the PCCs in table I, indicating that there are strong linear dependencies between the inputs and outputs of the preceptrons. Since most music signals are relatively stable, a linear combination of the input samples generally achieves a better interpolation accuracy than a nonlinear input composition.

F. Parameter Configurations

The optimal parameter configurations for both the noise detection and reconstruction ANNs are given in table III. The layers' structure is given in the format g-o, where g represents the number of input neurons and o the number of output neurons. The last column in the table is given in the format u-v. The batch trained ANNs performed best with a limit u of either 256 or 768 training patterns. These training patterns were delayed by v, which was set to six samples for the FSB-TDANN and the BSB-TDANN, and to eight samples for the other TDANNs.

IX. Empirical Analysis

This section presents the empirical result of the TDANNs for the outlier detection and reconstruction processes. The performance of the TDANNs is compared with that of other outlier detection and reconstruction algorithms discussed in [64] and [65] which are based on proximity-based algorithms, trigonometric approaches, polynomials and commonly used time series models. The algorithms in [64] and [65] are static and in most cases are unable to adapt to different signals changing over time. By employing ANNs, the detection and reconstruction algorithms adapt to the signal over time, providing more accurate predictions and interpolations of the signal. Finally, correlations are drawn between the results from the ANNs proposed in this article and those from other publications and a practical example is given.

A. Noise Detection

Previous research benchmarked a number of statistical outlier detection algorithms such as the standard score (SS), me-

TABLE III
The optimal parameter configurations for the ANNs.

Network Type	Layer Structure	Training Algorithm	Learning Rate	Activation Function	Training Epochs	Training Patterns
APD-TDANN	12-K	Incremental Backprop	0.1	Symmetric Elliot	1	-
APD-SRTDANN	17-1	Incremental Backprop	0.1	Hyperbolic Tangent	1	-
FI-TDANN	832-K	Incremental Backprop	0.1	Symmetric Elliot	1	-
BI-TDANN	832-K	Incremental Backprop	0.1	Symmetric Elliot	1	-
FI-SRTDANN	961-1	Incremental Backprop	0.1	Hyperbolic Tangent	1	-
BI-SRTDANN	961-1	Incremental Backprop	0.1	Hyperbolic Tangent	1	-
FSB-TDANN	224-k	iRprop$^-$	-	Symmetric Elliot	50	256-6
BSB-TDANN	224-k	iRprop$^-$	-	Symmetric Elliot	50	256-6
FCB-TDANN	220-K	iRprop$^-$	-	Symmetric Elliot	50	256-8
BCB-TDANN	220-K	iRprop$^-$	-	Symmetric Elliot	50	256-8
IB-TDANN	256-k	iRprop$^-$	-	Symmetric Elliot	50	768-8

dian absolute deviation (MAD), Mahalanobis distance (MHD), nearest neighbour deviation (NND) and the mean absolute spectral deviation (MASD) [64]. In addition, the APD algorithm was tested by predicting the input series using various polynomials and time series models. The polynomials included the standard polynomial (STP), osculating standard polynomial (OSP), Fourier polynomial (FOP), osculating Fourier polynomial (OFP), Newton polynomial (NEP) and the Hermite polynomial (HEP). The time series models consisted of the autoregressive (AR) model, moving average (MA) model, a combination of the AR and MA models (ARMA), AR integrated MA (ARIMA) model, AR conditional heteroskedasticity (ARCH) model and the generalized ARCH (GARCH) model.

Figure 6 shows the detection sensitivity for an increasing multivariate outlier duration. The figure compares the two TDANNs used in conjunction with the APD algorithm, with the other best performing outlier detectors. The TDANNs were only superior to the SS, MAD and MHD for most durations, but lagged behind the NND and the other APD-based algorithms which achieved a sensitivity of approximately 0.8 or higher.

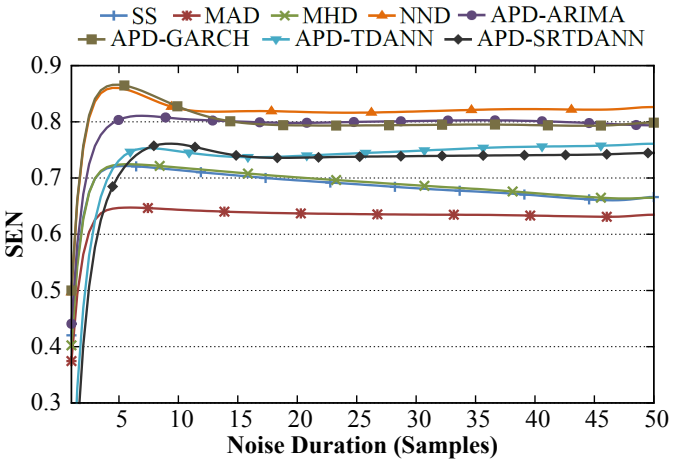

Fig. 6. The detection sensitivity for an increasing noise duration.

The detection accuracy for different genres is given in figure 7. The TDANNs' accuracy was slightly inferior to some of the other algorithms for stable signals, such as classical, country, jazz and rock music. Noise detection in more volatile signals, including electronic, metal, pop and reggae music, benefited more from the TDANNs. Since a TDANN learns the temporal characteristics of the input signal, it adapts to changes in the signal, which is problematic or even impossible with the other detection algorithms. The APD-GARCH also performed well for unstable signals, since it was specifically designed for high volatile financial markets [66]–[68].

The detailed comparison of the noise detection TDANNs and the outlier detectors from [64] is given in table IV. The highest sensitivity and specificity was achieved by the NND and MHD respectively. The best detection rate, as indicated by the MCC, was accomplished by employing the APD algorithm using an ARIMA model. The main cause of the slightly lower detection accuracy of the TDANNs compared to the AR,

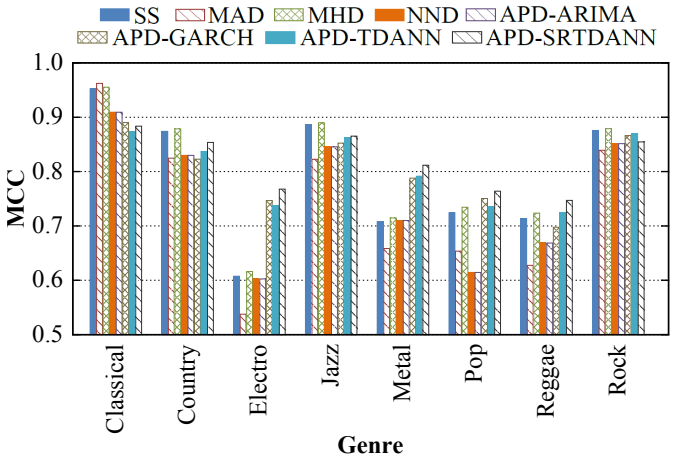

Fig. 7. The detection accuracy for different genres.

ARMA and ARIMA models was the more accurate prediction of the AR-based models for durations of seven samples and shorter. On average the TDANNs had a good forecasting accuracy for all durations, whereas the AR-based models were slightly superior with regards to the prediction of the immediate samples following the input data. The APD-TDANN had the lowest computational time. The TDANNs also achieved the highest tradeoff between the detection accuracy and the computational time with a SAT score twice as high as the next best algorithm. Due to a small number of neurons, the TDANN detectors were one of only four noise identification algorithms that were executed in real time. Although the TDANNs had a lower detection accuracy, the MCC difference of 0.009 between the APD-SRTDANN and APD-ARIMA algorithms is so marginal, that in practice most users would put more emphasis on the lower execution time. A detailed report on the performance of the outlier detection algorithms according to the noise duration and genre is given in appendix B.

TABLE IV
THE NOISE DETECTION SENSITIVITY, SPECIFICITY, ACCURACY, COMPUTATIONAL TIME AND TRADEOFF.

Algorithm	SEN	SPE	MCC	Time	SAT
SS	0.68113	0.99976	0.79238	5.4666	0.1405
MAD	0.63657	0.99937	0.74072	19.164	0.0430
MHD	0.68519	**0.99982**	0.79887	15.473	0.0533
NND	**0.82475**	0.99303	0.75386	1.6906	0.3503
MASD	0.38406	0.99834	0.55823	0.5064	0.4594
APD-STP	0.78489	0.99749	0.80177	13.806	0.0595
APD-OSP	0.78345	0.99762	0.80375	30.858	0.0273
APD-FOP	0.80210	0.99750	0.81840	24.692	0.0340
APD-OFP	0.80095	0.99743	0.82019	68.227	0.0125
APD-NEP	0.71231	0.99925	0.80804	7.1771	0.1104
APD-HEP	0.53563	0.99807	0.64657	0.4968	0.5410
APD-AR	0.81119	0.99802	0.83552	4.2692	0.1796
APD-MA	0.75252	0.99586	0.74685	24.767	0.0336
APD-ARMA	0.82254	0.99753	0.83456	26.644	0.0316
APD-ARIMA	0.80346	0.99826	**0.83668**	11.497	0.0715
APD-ARCH	0.80114	0.99795	0.82990	14.585	0.0568
APD-GARCH	0.80114	0.99795	0.82990	14.585	0.0568
APD-TDANN	0.75388	0.99774	0.78043	**0.1421**	1.0481
APD-SRTDANN	0.74422	0.99927	0.82722	0.1637	**1.2325**

B. Noise Reconstruction

Prior research analysed various duplication and trigonometric approaches, polynomials and time series models which showed potential for the reconstruction of audio signals but mostly lacked the ability to learn the temporal characteristics of the signal [65]. A number of duplication techniques were benchmarked, including the adjacent window interpolation (AWI), mirroring window interpolation (MWI), nearest neighbour interpolation (NNI) and similarity interpolation (SI). Cosine interpolation (CI) and Lanczos interpolation (LI) formed part of the trigonometric methods. Besides the STP, OSP, FOP, OFP, NEP and HEP, spline interpolation was also considered for the STP (SPS) and the FTP (FPS). The AR, MA, ARMA, ARIMA, ARCH and GARCH models were tested to determine the capacity to which they able to reconstruct audio signals.

Figure 8 illustrates the examined TDANNs' interpolation error for an increasing noise duration. The IB-TDANN performed best over all gap sizes, followed by the BI-SRTDANN.

The reconstruction accuracy for different genres is given in figure 9. The IB-TDANN achieved the lowest NRMSE for all genres, except for metal and rock music which was more accurately interpolated using the BI-SRTDANN.

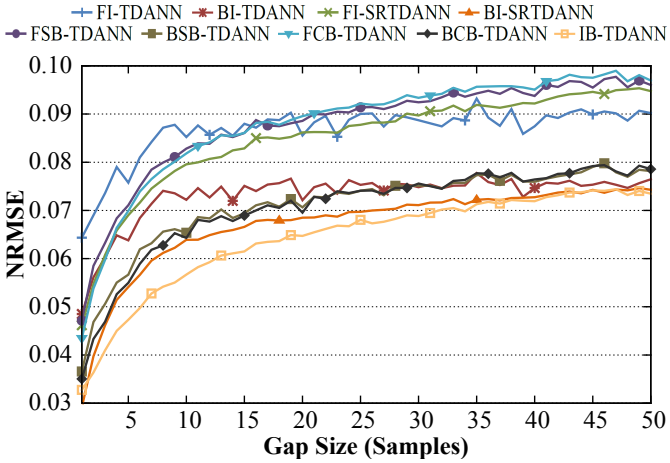

Fig. 8. The interpolation accuracy for an increasing noise duration.

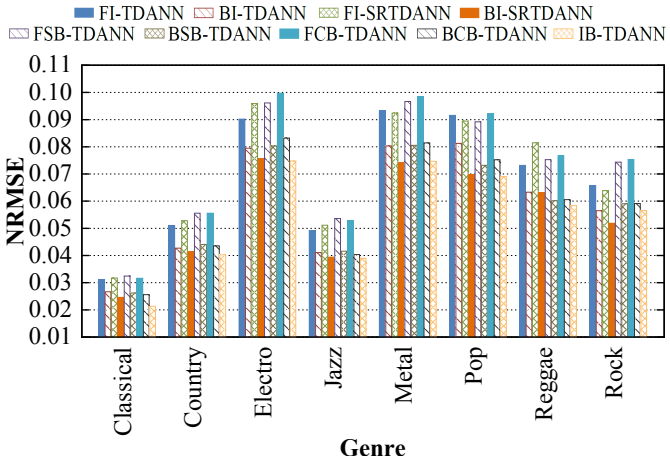

Fig. 9. The interpolation accuracy for different genres.

Table V lists the overall interpolation error, execution time and the tradeoff for the different reconstruction algorithms. The IB-TDANN performed best overall, followed by the BI-SRTDANN and BSB-TDANN. None of the TDANNs were able to execute in under 1 s\s using a single thread, therefore, making them infeasible for real time applications. However, since the digitization process of gramophone records is only conducted once for every recording, a higher audio quality is mostly preferred above real time execution. The full comparison of all interpolation algorithms according to the noise duration and genre is given in appendix C.

TABLE V
THE INTERPOLATION ACCURACY, COMPUTATIONAL TIME AND TRADEOFF.

Algorithm	NRMSE	Time	SAT
AWI	0.111371	0.049794	0.593457
MWI	0.104806	0.051691	0.628663
NNI	0.090748	0.027313	0.735624
SI	0.087269	0.054413	0.74839
LI	0.093215	0.027908	0.716238
CI	0.081218	**0.027128**	0.820088
STP	0.080057	0.03149	0.828606
OSP	0.086014	0.034523	0.770803
SPS	0.080057	0.038588	0.823627
FOP	0.122412	0.068523	0.535829
OFP	0.117923	5.325502	0.138812
FPS	0.081493	0.033035	0.813341
NEP	0.080058	0.027329	0.831552
HEP	0.081066	0.027557	0.821287
AR	0.071764	0.092778	**0.870951**
MA	0.087952	0.029541	0.757201
ARMA	0.071709	2.435243	0.281283
ARIMA	0.080201	6.808781	0.122321
ARCH	0.089057	0.062245	0.72967
GARCH	0.089057	0.062378	0.729598
FI-TDANN	0.068145	64.46232	0.014868
BI-TDANN	0.058917	129.051	0.00749
FI-SRTDANN	0.069853	2.663952	0.265745
BI-SRTDANN	0.054953	5.23738	0.161561
FSB-TDANN	0.071659	52.02565	0.018339
BSB-TDANN	0.058147	104.2376	0.009259
FCB-TDANN	0.072788	89.62463	0.010731
BCB-TDANN	0.058637	179.7895	0.005386
IB-TDANN	**0.054247**	100.0583	0.009648

C. State of the Art Comparison

All ANNs proposed in this article had a considerable improvement in the interpolation accuracy over the ANN proposed by Czyzewski [16]. In addition, the time required to train the ANNs was substantially lower in comparison to the training time of Czyzewski's approach, especially for the recurrent architectures.

The proposed ANNs had a reconstruction accuracy comparable to that of the methods proposed by Cocchi and Uncini [18]. The best performing architecture in this paper, namely the IB-TDANN, had an interpolation NRMSE between 0.020 and 0.058. In comparison, Cocchi's best performing ANN only achieved a NRMSE of approximately 0.024 to 0.105. Although, it has to be noted that Cocchi reconstructed larger gap sizes which has a direct impact on the interpolation and justifies the larger error. No indication was given in relation to the training time. However, since Cocchi used hidden layers,

more neurons and utilized backpropagation for training, which generally has a slower convergence than iRprop, training in [18] is computationally more expensive.

It is also important to note that different datasets were used for the experiments in the various articles, which might also had an impact the the reconstruction accuracy. A very small dataset was used in [16] and [18] which does not necessarily represent the performance for a variety of songs, especially from different genres. For instance, Cocchi only used a few songs from the classical, jazz, and pop genres [18], which are stable and easier to reconstruct compared to more volatile genres such as metal and electro, as shown in figure 9.

D. Reconstruction Example

Figure 10 depicts an example of a sound wave before and after reconstruction using the APD-SRTDANN for the noise detection and the IB-TDANN for the interpolation. The signal in question is a segment from Mozart's "Eine kleine Nachtmusik". All the impulse disturbances where successfully removed and a reconstruction NRMSE of 0.042 was achieved.

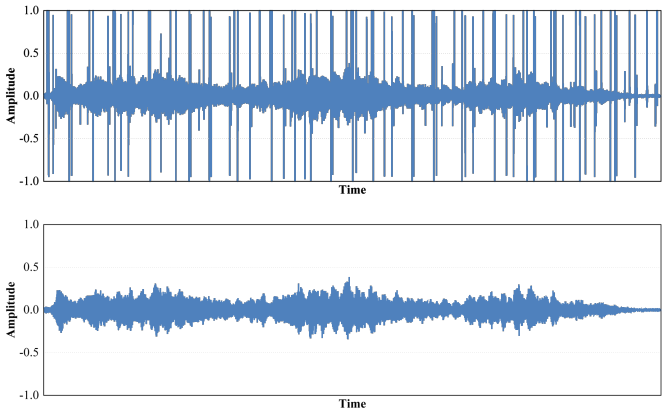

Fig. 10. The sound wave before and after reconstruction.

X. Conclusion

Analogue storage mediums, such as gramophone records, are subject to noise caused by damages to the medium. This article investigated a number of TDANNs for the detection and reconstruction of distortions in gramophone audio signals. The TDANNs were compared to other algorithms, including polynomials and time series models amongst others. It was found that the TDANNs for outlier detection purposes were slightly inferior to other detection algorithms, but were more efficient by achieving an acceptable detection rate within a limited timespan. The interpolation TDANNs had a notable decrease in the reconstruction error, but were computationally expensive due to training a network with a large number of neurons.

This article processed the channels in audio signal separately. Future research should examine the combined processing of all available channels using TDANNs. The noise from one channel can, therefore, be detected and reconstructed using the information from other channels. In addition, due to the performance deviation between different genres, further investigation is needed to determine the capabilities of a TDANN when trained in a dynamic environment. A TDANN might achieve a better detection and interpolation rate if it dynamically adapts to the current volatility of the signal. One set of parameters may be used when the signal is stable, and another one if the volatility increases.

This article also directly used the time series data as input to the TDANNs. A more accurate approximation of the signal may be achieved by generating the ANNs' training patterns with the output of other time series models, such as the AR, ARIMA or GARCH model. Combing the output of other models with the time series data and feeding it as input into an ANN or using other ANNs other than TDANNs may also prove beneficial.

The training time of the reconstruction ANNs is still high, which is a common problem with most ANNs [16], [17]. Further research is needed in order to determine how the computational time can be reduced.

Appendix A
Network Structures

Figure 11 - 17 show the ANN structures as discussed in section IV and section V. The vertical arrows on the left indicate the direction in which the consecutive patterns are presented to the ANNs as the sample window moves across the signal.

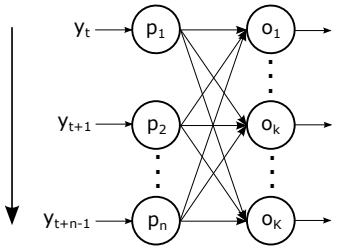

Fig. 11. The network structure of the APD-TDANN, FI-TDANN, BI-TDANN, FCB-TDANN, and the BCB-TDANN.

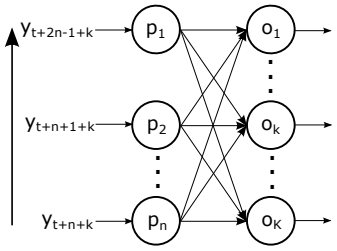

Fig. 12. The network structure of the BI-TDANN and the BCB-TDANN.

Appendix B
Detection Algorithms

Table VI shows the detection sensitivity for an increasing noise duration. The APD employing the ARCH and GARCH

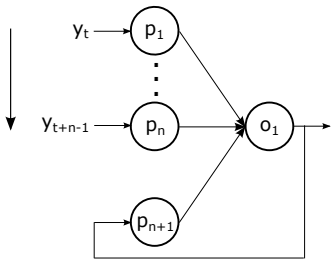

Fig. 13. The network structure of the APD-SRTDANN, FI-SRTDANN, and the BI-SRTDANN.

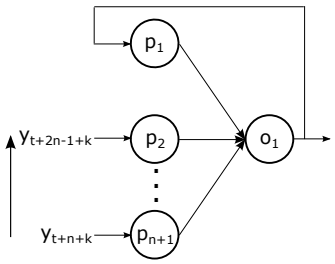

Fig. 14. The network structure of the BI-SRTDANN.

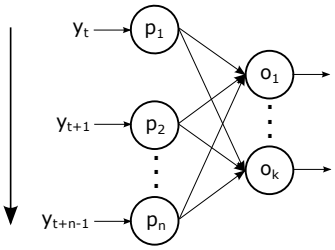

Fig. 15. The network structure of the FSB-TDANN.

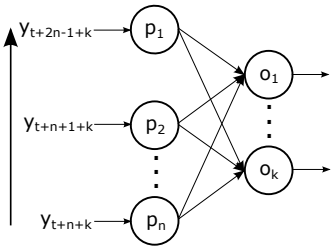

Fig. 16. The network structure of the BSB-TDANN.

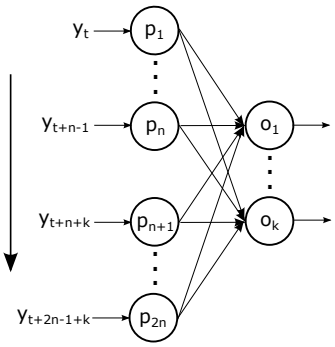

Fig. 17. The network structure of the IB-TDANN.

models detected most outliers of ten samples or shorter. The NND had the best detection sensitivity for all other noise durations.

Table VII shows the detection accuracy for different genres. The TDANNs had a good detection rate for the volatile signals, with the APD-SRTDANN achieving the highest MCC for electronic music.

Appendix C
Reconstruction Algorithms

Table VIII shows the interpolation performance of all reconstruction algorithms for an increasing gap size. The ARIMA model outperformed all TDANNs for durations of five samples or shorter. The IB-TDANN interpolated most accurately for all other gap sizes.

Table IX provides the detailed reconstruction performance for different genres. The IB-TDANN performed best for all genres, except for metal and rock music, which had a better interpolation using the BI-SRTDANN.

References

[1] R. R. Wile, "Etching the Human Voice: The Berliner Invention of the Gramophone," *Association for Recorded Sound Collections Journal*, vol. 21, no. 1, pp. 2–22, 1990.

[2] J. Attali, *Noise: The Political Economy of Music*, ser. Theory and History of Literature. Manchester, United Kingdom: Manchester University Press, 1985.

[3] F. Richter, "The LP is Back!" http://www.statista.com/chart/1465/vinyl-lp-sales-in-the-us, January 2014, accessed: 2014-04-16.

[4] C. M. Update, "SoundScan may be under reporting US vinyl sales," http://www.thecmuwebsite.com/article/soundscan-may-be-under-reporting-us-vinyl-sales, October 2011, accessed: 2014-04-16.

[5] P. Rayan Kutty and A. Sreenivasa Murthy, "Kalman Filter Using Quantile Based Noise Estimation for Audio Restoration," in *International Conference on Emerging Trends in Electrical and Computer Technology*, 2011, pp. 616–620.

[6] S. Herzog, "Efficient DSP Implementation of Median Filtering for Real-Time Audio Noise Reduction," in *International Conference on Digital Audio Effects*, 2013, pp. 1–6.

[7] S. Godsill, P. Rayner, and O. Capp, "Digital Audio Restoration," in *Applications of Digital Signal Processing to Audio and Acoustics*, ser. The International Series in Engineering and Computer Science, 2002, vol. 437, pp. 133–194.

[8] A. Uncini, "Audio Signal Processing by Neural Networks," *Neurocomputing*, vol. 55, no. 3-4, pp. 593–625, 2003.

[9] V. Peddinti, D. Povey, and S. Khudanpur, "A Time Delay Neural Network Architecture for Efficient Modeling of Long Temporal Contexts," in *Proceedings of Interspeech*, 2015.

[10] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30–42, 2012.

[11] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "An Experimental Study on Speech Enhancement Based on Deep Neural Networks," *Signal Processing Letters, IEEE*, vol. 21, no. 1, pp. 65–68, 2014.

[12] H. C. C. Carneiro, F. M. G. França, and P. M. V. Lima, "Multilingual Part-of-speech Tagging with Weightless Neural Networks," *Neural Networks*, vol. 66, pp. 11–21, 2015.

[13] M. Ravanelli, B. Elizalde, K. Ni, and G. Friedland, "Audio Concept Classification with Hierarchical Deep Neural Networks," in *Proceedings of the European Signal Processing Conference*, 2014, pp. 606–610.

[14] S. Sigtia and S. Dixon, "Improved Music Feature Learning with Deep Neural Networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014, pp. 6959–6963.

[15] B. Defraene, N. Mansour, S. De Hertogh, T. van Waterschoot, M. Diehl, and M. Moonen, "Declipping of Audio Signals Using Perceptual Compressed Sensing," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 12, pp. 2627–2637, 2013.

TABLE VI
THE NOISE DETECTION SENSITIVITY FOR DIFFERENT NOISE DURATIONS.

Algorithm	Noise Duration (Samples)									
	1-5	6-10	11-15	16-20	21-25	26-30	31-35	36-40	41-45	46-50
SS	0.669583	0.742467	0.689516	0.720873	0.670295	0.708234	0.654351	0.696723	0.642453	0.683814
MAD	0.599158	0.667134	0.623614	0.656070	0.618275	0.653976	0.615453	0.652637	0.614562	0.649319
MHD	0.668511	0.746199	0.694955	0.725869	0.675914	0.712327	0.661077	0.699641	0.647905	0.684411
NND	0.800853	0.874539	**0.780308**	**0.856599**	**0.780291**	**0.852318**	**0.783123**	**0.858536**	**0.787618**	**0.857462**
MASD	0.015257	0.143573	0.296753	0.310271	0.370858	0.403118	0.404717	0.407955	0.423841	0.433945
APD-STP	0.713340	0.802340	0.730461	0.798646	0.740985	0.810642	0.746562	0.819477	0.757231	0.824678
APD-OSP	0.743536	0.821894	0.740147	0.800336	0.739647	0.808002	0.743992	0.816667	0.751706	0.819986
APD-FOP	0.802755	0.879977	0.768096	0.835465	0.757905	0.834036	0.756318	0.835529	0.755607	0.834490
APD-OFP	0.803470	0.880603	0.766428	0.833854	0.754655	0.834256	0.754219	0.835929	0.754867	0.833960
APD-NEP	0.711705	0.797715	0.665926	0.747894	0.660226	0.752640	0.657973	0.751863	0.656937	0.753587
APD-HEP	0.417349	0.586179	0.564707	0.566440	0.503448	0.552414	0.500766	0.554893	0.503621	0.554214
APD-AR	0.751333	0.860521	0.777807	0.843201	0.770887	0.844394	0.772310	0.845398	0.768000	0.840682
APD-MA	0.742909	0.813874	0.715799	0.777359	0.710079	0.779870	0.710877	0.784304	0.714342	0.784997
APD-ARMA	0.751686	0.860600	0.777932	0.843096	0.770672	0.844120	0.772229	0.845113	0.767624	0.840655
APD-ARIMA	0.718782	0.843681	0.771223	0.832146	0.763298	0.836598	0.766161	0.838789	0.763610	0.831715
APD-ARCH	**0.805593**	**0.883358**	0.768265	0.834933	0.752735	0.834458	0.753625	0.836317	0.752317	0.835253
APD-GARCH	**0.805593**	**0.883358**	0.768265	0.834933	0.752735	0.834458	0.753625	0.836317	0.752317	0.835253
APD-TDANN	0.560256	0.782041	0.712988	0.769727	0.706188	0.782426	0.713407	0.792590	0.719061	0.796397
APD-SRTDANN	0.503477	0.787513	0.720407	0.773107	0.694871	0.781423	0.696044	0.785262	0.695034	0.788981

TABLE VII
THE NOISE DETECTION ACCURACY (MCC) FOR DIFFERENT GENRES.

Algorithm	Classical	Country	Electro	Jazz	Metal	Pop	Reggae	Rock
SS	0.951926	0.873877	0.607356	0.886052	0.707332	0.724012	0.713421	0.875098
MAD	**0.962271**	0.824588	0.537702	0.822593	0.658377	0.653664	0.627439	0.839129
MHD	0.955160	0.878473	0.616193	**0.889799**	0.714632	0.734291	0.723507	0.878881
NND	0.909140	0.829856	0.602421	0.845456	0.709829	0.614396	0.668494	0.851308
MASD	0.608033	0.598760	0.521788	0.595016	0.541185	0.492498	0.512974	0.595583
APD-STP	0.890176	0.822822	0.746661	0.852363	0.787955	0.750420	0.697524	0.866243
APD-OSP	0.873373	0.837015	0.737313	0.861959	0.791422	0.734935	0.724041	0.869901
APD-FOP	0.883528	0.853576	0.767727	0.865233	0.811626	0.763926	0.747050	0.854538
APD-OFP	0.895254	0.860299	0.763816	0.859296	0.806712	0.765350	0.727298	**0.883528**
APD-NEP	0.884826	0.863891	0.790151	0.855927	**0.843132**	**0.819096**	**0.804737**	0.860507
APD-HEP	0.917257	0.660361	0.516832	0.652523	0.631021	0.528816	0.600382	0.665352
APD-AR	0.954877	0.879215	0.776779	0.878765	0.808909	0.776253	0.756497	0.852870
APD-MA	0.874753	0.842858	0.592617	0.848724	0.707013	0.621438	0.660757	0.826653
APD-ARMA	0.954944	0.877964	0.776087	0.879280	0.806389	0.779852	0.753114	0.848835
APD-ARIMA	0.956025	**0.889603**	0.759254	0.879509	0.813750	0.775763	0.755577	0.863971
APD-ARCH	0.899874	0.864925	0.779141	0.863870	0.828926	0.781369	0.756995	0.864109
APD-GARCH	0.899874	0.864925	0.779141	0.863870	0.828926	0.781369	0.756995	0.864109
APD-TDANN	0.887571	0.832427	0.683407	0.830236	0.759126	0.702497	0.716537	0.831674
APD-SRTDANN	0.912702	0.837436	**0.790940**	0.832761	0.815509	0.800253	0.793073	0.835042

[16] A. Czyzewski, "Learning Algorithms for Audio Signal Enhancement, Part 1: Neural Network Implementation for the Removal of Impulse Distortions," *Journal of the Audio Engineering Society*, vol. 45, no. 10, pp. 815–831, 1997.

[17] A. Czyzewski and R. Krolikowski, "Neuro-rough Control of Masking Thresholds for Audio Signal Enhancement," *Neurocomputing*, vol. 36, no. 4, pp. 5–27, 2001.

[18] G. Cocchi and A. Uncini, "Subband Neural Networks Prediction for On-line Audio Signal Recovery," *IEEE Transactions on Neural Networks*, vol. 13, no. 4, pp. 867–876, 2002.

[19] K. P. Seng, L. E. Hui, and T. Ming, "Multimedia Signal Processing using AI," in *Asia-Pacific Conference on Communications*, vol. 2, 2003, pp. 825–829.

[20] S. J. Godsill and P. J. Rayner, *Digital Audio Restoration - A Statistical Model-Based Approach*, 1st ed., London, UK, 1998.

[21] S. V. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*, 4th ed. Hoboken, New Jersey, USA: Wiley, 2009.

[22] D. J. Hill and B. S. Minsker, "Anomaly Detection in Streaming Environmental Sensor Data: A Data-driven Modeling Approach," *Environmental Modelling and Software*, vol. 25, no. 9, pp. 1014–1022, 2010.

[23] M. Niedźwiecki and M. Ciołek, "Elimination of Clicks from Archive Speech Signals Using Sparse Autoregressive Modeling," in *Proceedings of the 20th European Signal Processing Conference*, August 2012, pp. 2615–2619.

[24] R. S. Tsay, "Outliers, Level Shifts, and Variance Changes in Time Series," *Journal of Forecasting*, vol. 7, no. 1, pp. 1–20, 1988.

[25] J. Ma and S. Perkins, "Online Novelty Detection on Temporal Sequences," in *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*, 2003, pp. 613–618.

[26] M. C. Hau and H. Tong, "A Practical Method for Outlier Detection in Autoregressive Time Series Modelling," *Stochastic Hydrology and Hydraulics*, vol. 3, no. 4, pp. 241–260, 1989.

[27] K. J. Lang, A. H. Waibel, and G. E. Hinton, "A Time-delay Neural Network Architecture for Isolated Word Recognition," *Neural Networks*, vol. 3, no. 1, pp. 23–43, 1990.

[28] H. Ge, W. Du, F. Qian, and Y. Liang, "A Novel Time-delay Recurrent Neural Network and Application for Identifying and Controlling Nonlinear Systems," in *International Conference on Natural Computation*, vol. 1, 2007, pp. 44–48.

[29] C. L. Giles, S. Lawrence, and A. C. Tsoi, "Noisy time series prediction using recurrent neural networks and grammatical inference," *Machine Learning*, vol. 44, no. 1-2, pp. 161–183, 2001.

[30] S.-S. Kim, "Time-delay Recurrent Neural Network for Temporal Correlations and Prediction," *Neurocomputing*, vol. 20, no. 1-3, pp. 253–263, 1998.

[31] M. I. Jordan, "Attractor Dynamics and Parallelism in a Connectionist Sequential Machine," in *Proceedings of the Cognitive Science Society*, 1986, pp. 531–546.

TABLE VIII
THE INTERPOLATION ACCURACY (NRMSE) FOR DIFFERENT NOISE DURATIONS.

Algorithm	Noise Duration (Samples)									
	1-5	6-10	11-15	16-20	21-25	26-30	31-35	36-40	41-45	46-50
AWI	0.061014	0.109485	0.125043	0.136948	0.144415	0.150030	0.153123	0.154301	0.156253	0.155272
MWI	0.059964	0.103136	0.119742	0.130193	0.135851	0.140522	0.143783	0.144848	0.146167	0.146004
NNI	0.056626	0.079500	0.093445	0.103867	0.110910	0.116589	0.121023	0.124764	0.128621	0.131256
SI	0.046737	0.077438	0.091772	0.101091	0.107379	0.112918	0.117000	0.120439	0.123543	0.125800
LI	0.082024	0.090207	0.098707	0.108305	0.115674	0.122870	0.126692	0.128696	0.130498	0.130745
CI	0.040126	0.065559	0.079875	0.090035	0.097060	0.103097	0.107760	0.111767	0.115634	0.118414
STP	0.039622	0.064877	0.079025	0.088880	0.095711	0.101678	0.106250	0.110211	0.113940	0.116665
OSP	0.061543	0.081832	0.092446	0.100407	0.105845	0.110766	0.114665	0.117806	0.121388	0.123443
SPS	0.039622	0.064877	0.079025	0.088880	0.095711	0.101678	0.106250	0.110211	0.113940	0.116665
FOP	0.202796	0.186715	0.178200	0.172345	0.168306	0.165825	0.163481	0.161402	0.160532	0.159072
OFP	0.184722	0.172651	0.166623	0.162677	0.160186	0.158954	0.157642	0.156505	0.156499	0.155782
FPS	0.078763	0.086066	0.092466	0.097982	0.101775	0.105398	0.108191	0.110807	0.113462	0.115133
NEP	0.039622	0.064877	0.079025	0.088880	0.095711	0.101678	0.106250	0.110211	0.113940	0.116665
HEP	0.039987	0.065400	0.079712	0.089849	0.096861	0.102896	0.107556	0.111562	0.115421	0.118198
MA	0.054773	0.081479	0.094212	0.102923	0.108755	0.114096	0.117949	0.121148	0.124229	0.126381
AR	0.030029	0.057569	0.071434	0.080595	0.086715	0.091646	0.095414	0.098726	0.101605	0.103628
ARMA	0.030004	0.057530	0.071380	0.080535	0.086644	0.091576	0.095325	0.098638	0.101511	0.103538
ARIMA	**0.029804**	0.058760	0.074316	0.085561	0.093578	0.100808	0.106410	0.111270	0.115704	0.119112
ARCH	0.066648	0.088248	0.098274	0.106027	0.111099	0.116074	0.119463	0.122308	0.125319	0.127295
GARCH	0.066648	0.088248	0.098274	0.106027	0.111099	0.116074	0.119463	0.122308	0.125319	0.127295
FI-TDANN	0.072333	0.085071	0.086784	0.088125	0.088401	0.089063	0.089311	0.088237	0.090019	0.090038
BI-TDANN	0.058700	0.071916	0.073863	0.074745	0.075087	0.075114	0.075603	0.075026	0.075571	0.075606
FI-SRTDANN	0.059302	0.076061	0.081416	0.085289	0.086782	0.088947	0.091104	0.091821	0.093910	0.094866
BI-SRTDANN	0.044118	0.060706	0.065364	0.068048	0.069115	0.070546	0.071822	0.072469	0.073727	0.074216
FSB-TDANN	0.061664	0.079459	0.084955	0.088101	0.090425	0.091868	0.093698	0.094503	0.096128	0.096704
BSB-TDANN	0.049145	0.064435	0.068994	0.071298	0.073352	0.074577	0.075689	0.076588	0.077611	0.078249
FCB-TDANN	0.058659	0.078179	0.084913	0.088521	0.091095	0.092803	0.094748	0.095571	0.097449	0.097802
BCB-TDANN	0.046572	0.062613	0.068248	0.070626	0.073378	0.074491	0.075936	0.076933	0.077963	0.078512
IB-TDANN	0.040480	**0.053674**	**0.060134**	**0.063966**	**0.066642**	**0.068205**	**0.070240**	**0.071853**	**0.073499**	**0.073833**

TABLE IX
THE INTERPOLATION ACCURACY (NRMSE) FOR DIFFERENT GENRES.

Algorithm	Classical	Country	Electro	Jazz	Metal	Pop	Reggae	Rock
AWI	0.094831	0.112448	0.119919	0.117352	0.112000	0.111454	0.122570	0.100393
MWI	0.082085	0.100996	0.114205	0.105853	0.107922	0.105229	0.110541	0.094273
NNI	0.065920	0.085024	0.105180	0.087931	0.103765	0.094996	0.098086	0.085083
SI	0.067175	0.083507	0.099116	0.086747	0.095450	0.090004	0.095424	0.080727
LI	0.064791	0.085723	0.111587	0.089531	0.105977	0.099367	0.101492	0.087248
CI	0.058249	0.075599	0.094691	0.077718	0.093754	0.085322	0.088126	0.076281
STP	0.057997	0.074762	0.093071	0.076897	0.091734	0.083901	0.087085	0.075014
OSP	0.065010	0.082054	0.097175	0.084630	0.096641	0.088736	0.093376	0.080489
SPS	0.057997	0.074762	0.093071	0.076897	0.091734	0.083901	0.087085	0.075014
FOP	0.085755	0.118242	0.141207	0.123283	0.137256	0.132658	0.122470	0.118423
OFP	0.084521	0.113369	0.136857	0.118546	0.130239	0.126993	0.120449	0.112408
FPS	0.056885	0.074597	0.097833	0.077936	0.093035	0.086956	0.088596	0.076109
NEP	0.057997	0.074762	0.093071	0.076897	0.091734	0.083901	0.087085	0.075014
HEP	0.058174	0.075469	0.094503	0.077584	0.093540	0.085153	0.087980	0.076127
MA	0.068295	0.084632	0.099053	0.087845	0.095938	0.090286	0.096153	0.081411
AR	0.044684	0.063066	0.089517	0.063907	0.083930	0.079821	0.079634	0.069552
ARMA	0.044558	0.063004	0.089505	0.063807	0.083903	0.079800	0.079607	0.069484
ARIMA	0.057269	0.074376	0.096327	0.075326	0.087477	0.085975	0.090019	0.074839
ARCH	0.070394	0.086568	0.099060	0.089795	0.096041	0.090729	0.097582	0.082284
GARCH	0.070394	0.086568	0.099060	0.089795	0.096041	0.090729	0.097582	0.082284
FI-TDANN	0.031090	0.051052	0.090200	0.049207	0.093168	0.091573	0.073144	0.065724
BI-TDANN	0.026660	0.042709	0.079468	0.041094	0.080332	0.081278	0.063294	0.056501
FI-SRTDANN	0.031740	0.052813	0.095891	0.051151	0.092425	0.089477	0.081454	0.063870
BI-SRTDANN	0.024608	0.041309	0.075553	0.039394	**0.074140**	0.069712	0.063043	**0.051867**
FSB-TDANN	0.032444	0.055585	0.096158	0.053605	0.096645	0.089199	0.075290	0.074350
BSB-TDANN	0.026225	0.044085	0.080297	0.041578	0.080555	0.073196	0.060170	0.059070
FCB-TDANN	0.031616	0.055558	0.099710	0.052884	0.098522	0.092059	0.076620	0.075334
BCB-TDANN	0.025598	0.043566	0.083219	0.040344	0.081443	0.075228	0.060567	0.059130
IB-TDANN	**0.021309**	**0.040269**	**0.074772**	**0.038974**	0.074652	**0.069089**	**0.058380**	0.056536

[32] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, "Using Recurrent Neural Networks for Slot Filling in Spoken Language Understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, 2015.

[33] E. Marchi, G. Ferroni, F. Eyben, L. Gabrielli, S. Squartini, and B. Schuller, "Multi-resolution Linear Prediction Based Features for Audio Onset Detection with Bidirectional LSTM Neural Networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2014, pp. 2164–2168.

[34] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, November 1997.

[35] A. Atakulreka and D. Sutivong, "Avoiding Local Minima in Feedforward Neural Networks by Simultaneous Learning," in *Advances in Artificial Intelligence*. Springer, 2007, vol. 4830, pp. 100–109.

[36] R. Adhikari and R. Agrawal, "A Homogeneous Ensemble of Artificial Neural Networks for Time Series Forecasting," *International Journal of Computer Applications*, vol. 32, no. 7, 2013.

[37] A. Rakitianskaia and A. Engelbrecht, "Training Neural Networks with PSO in Dynamic Environments," in *IEEE Congress on Evolutionary Computation*, May 2009, pp. 667–673.

[38] L. Oudre, "Automatic Detection and Removal of Impulsive Noise in Audio Signals," *Image Processing On Line*, 2014.

[39] P. A. A. Esquef and G. S. Welter, "Audio De-thumping using Huang's Emperical Mode Decomposition," in *International Conference on Digital Audio Effects*, vol. 11, Paris, France, 2011, pp. 401–408.

[40] J. Howarth and P. J. Wolfe, "Correction of Wow and Flutter Effects in Analog Tape Transfers," in *Audio Engineering Society Convetion*, vol. 117, San Francisco, Californis, United States of America, 2011.

[41] M. Niedźwiecki and K. Cisowski, "Adaptive Scheme for Elimination of Background Noise and Impulsive Disturbances from Audio Signals," in *Quatrozieme Colloque*, Juan-les-Pins, France, 1993, pp. 519–522.

[42] K. Pearson, "Note on Regression and Inheritance in the Case of Two Parents," *Proceedings of the Royal Society of London*, vol. 58, no. 347, pp. 240–242, 1895.

[43] F. Galton, "Regression Towards Mediocrity in Hereditary Stature," *The Journal of the Anthropological Institute of Great Britain and Ireland*, vol. 15, pp. 246–263, 1886.

[44] B. W. Matthews, "Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme," *Biochimica et Biophysica Acta Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[45] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard, "Managing Performance vs Accuracy Trade-offs with Loop Perforation," in *Proceedings of the SIGSOFT Symposium and the European Conference on Foundations of Software Engineering*, 2011, pp. 124–134.

[46] R. F. Gunst and R. L. Mason, "Fractional Factorial Design," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 1, no. 2, pp. 234–244, 2009.

[47] P. Sibi, S. A. Jones, and S. P., "Analysis of Different Activation Functions Using Back Propagation Neural Networks," *Journal of Theoretical and Applied Information Technology*, vol. 47, no. 3, pp. 1264–1268, 2013.

[48] D. L. Elliot, "A Better Activation Function for Artificial Neural Networks," Institute for Systems Research, Tech. Rep., 1993, t.R. 93-8.

[49] P. J. Werbos, "Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences," Ph.D. dissertation, Harvard University, Cambridge, Massachusetts, United States of America, 1974.

[50] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, "Learning Representations by Back-propagating Errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[51] H. Darjazini, Q. Cheng, and R. Liyana-Pathirana, "Incremental Learning Algorithm for Speech Recognition," in *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering*. Springer, 2007, pp. 231–235.

[52] M. T. Vo, "Incremental Learning Using the Time Delay Neural Network," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, April 1994, pp. 629–632.

[53] M. Sugiyama, H. Sawai, and A. H. Waibel, "Review of TDNN Architectures for Speech Recognition," in *IEEE International Sympoisum on Circuits and Systems*, vol. 1, 1991, pp. 582–585.

[54] H. Sawai, "TDNN-LR Continuous Speech Recognition System Using Adaptive Incremental TDNN Training," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, 1991, pp. 53–56.

[55] S. E. Fahlman, "Faster-Learning Variations on Back-Propagation: An Empirical Study," in *Proceedings of the Connectionist Models Summer School*. Morgan-Kaufmann, 1988, pp. 38–51.

[56] M. Riedmiller and H. Braun, "RPROP - A Fast Adaptive Learning Algorithm," in *International Symposium on Computer and Information Sciences*, 1992, pp. 279–285.

[57] M. Riedmiller and H. Braun, "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm," in *IEEE International Conference on Neural Networks*, 1993, pp. 586–591.

[58] C. Igel and M. Hüsken, "Empirical Evaluation of the Improved Rprop Learning Algorithm," *Neurocomputing*, vol. 50, pp. 105–123, 2003.

[59] C. Igel and M. Hüsken, "Improving the Rprop Learning Algorithm," in *International Symposium on Neural Computation*, 2000, pp. 115–121.

[60] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, 2nd ed. Wiley Publishing, 2007.

[61] K. Hornik, M. B. Stinchcombe, and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[62] K. Hornik, M. B. Stinchcombe, and H. White, "Universal Approximation of an Unknown Mapping and its Derivatives using Multilayer Feedforward Networks," *Neural Networks*, vol. 3, no. 5, pp. 551–560, 1990.

[63] E. K. Blum and L. K. Li, "Approximation Theory and Feedforward Networks," *Neural Networks*, vol. 4, no. 4, pp. 511–515, 1991.

[64] C. F. Stallmann and A. P. Engelbrecht, "Digital Noise Detection in Gramophone Recordings," in *ACM International Conference on Multimedia*, Brisbane, Australia, 2015, in review.

[65] C. F. Stallmann and A. P. Engelbrecht, "Gramophone Noise Reconstruction: A Comparative Study of Interpolation Algorithms for Noise Reduction," in *International Conference on Signal Processing and Multimedia Applications*, 2015, in review.

[66] R. F. Engle, "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation," *Econometrica*, vol. 50, no. 4, pp. 987–1007, July 1982.

[67] T. Bollerslev, "Generalized Autoregressive Conditional Heteroskedasticity," *Journal of Econometrics*, vol. 31, no. 3, pp. 307–327, 1986.

[68] C. Francq and J.-M. Zakoian, *GARCH Models: Structure, Statistical Inference and Financial Applications*. Hoboken, New York, United States of America: Wiley, 2010.

Christoph F. Stallmann Christoph Stallmann received the BIT (Honours) degree in 2012 and is currently completing the Masters degree in Computer Science at the University of Pretoria, South Africa. He has worked at the Council for Scientific and Industrial Research (CSIR) and the South African National Space Agency (SANSA) and has been an assistant lecturer in Computer Science at the University of Pretoria since 2011. His research interests include music and audio analysis, digital signal processing, signal modelling, artificial neural networks, remote sensing, and geographic information systems. His current research focuses on the digital noise detection and reconstruction of gramophone audio recordings using polynomials, time series models and artificial neural networks.

Andries P. Engelbrecht Andries Engelbrecht received the Masters and PhD degrees in Computer Science from the University of Stellenbosch, South Africa, in 1994 and 1999 respectively. He is a Professor in Computer Science at the University of Pretoria, and serves as Head of the department. He also holds the position of South African Research Chair in Artificial Intelligence, and leads the Computational Intelligence Research Group at the University of Pretoria, consisting of 40 Masters and PhD students. His research interests include swarm intelligence, evolutionary computation, artificial neural networks, artificial immune systems, and the application of these Computational Intelligence paradigms to data mining, games, bioinformatics, finance, and difficult optimization problems. He has published over 220 papers in these fields in journals and international conference proceedings, and is the author of two books, Computational Intelligence: An Introduction and Fundamentals of Computational Swarm Intelligence.