An end-to-end spoofing countermeasure for automatic speaker verification using evolving recurrent neural networks

Giacomo Valenti[1,2], *Héctor Delgado*[2], *Massimiliano Todisco*[2], *Nicholas Evans*[2] *and Laurent Pilati*[1]

[1]NXP Semiconductors, Mougins, France ; [2]EURECOM, Biot, France

valenti@eurecom.fr, evans@eurecom.fr, delgado@eurecom.fr
todisco@eurecom.fr, laurent.pilati@nxp.com

Abstract

Research in anti-spoofing for automatic speaker verification has advanced considerably in the last three years. Anti-spoofing is a particularly difficult pattern classification problem since the characteristics of spoofed speech vary considerably and can never be predicted with any certainty in the wild. The design of features suited to the detection of unpredictable spoofing attacks is thus a staple of current research. End-to-end approaches to spoofing detection with exploit automatic feature learning have shown success and offer obvious appeal. This paper presents our efforts to develop such a system using recurrent neural networks and a particular algorithm known as neuroevolution of augmenting topologies (NEAT). Contributions include a new fitness function for network learning that not only results in better generalisation than the baseline system, but which also improves on raw performance by 22% relative when assessed using the ASVspoof 2017 database of bona fide speech and replay spoofing attacks. Results also show that mini-batch training helps to improve generalisation, a technique which could also be of benefit to other solutions to the spoofing detection problem.

1. Introduction

Automatic speaker verification (ASV) [1, 2] offers a convenient, reliable and cost-effective approach to person authentication. Voice-based authentication is nowadays used in a plethora of logical and physical access scenarios, e.g. for telephone banking or for smartphone logon [3]. Despite the success, vulnerabilities to spoofing (also known as presentation attacks) give reason for caution. Without adequate countermeasures, fraudsters can manipulate the normal operation of an authentication system by masquerading as genuine users and hence gain unauthorised access to protected resources or services. Vulnerabilities to presentation attacks are clearly inadmissible; in addition to the immediate security concerns, they undermine confidence in ASV technology.

It is known that ASV systems can be vulnerable to spoofing attacks in the form of impersonation, synthetic speech, converted voice and replay [4]. Impersonation (the imitation of a target speaker by another person) requires a certain skill and is generally considered to pose only a modest risk [5]. While the threats posed by synthetic speech and converted voice are potentially severe, given that their implementation requires specialist expertise, the actual risk may be relatively low. Replay attacks arguably present the greatest threat. Replay attacks involve the (surreptitious) capture and subsequent playback to the ASV system of a speech sample captured from a genuine speaker/user. The threat and risk posed by replay attacks is significant: replay attacks can be mounted easily with widely available, consumer-grade audio recording and playback devices (e.g. smart phones) and can be especially difficult to distinguish from genuine, bona fide speech samples.

Efforts to develop spoofing countermeasures, also known as presentation attack detection (PAD) systems, are now well under way; the study of spoofing countermeasures for ASV is today an established area of research [6]. The first competitive evaluation, namely the ASV spoofing and countermeasures (ASVspoof) challenge [7], was held in 2015. It promoted the development of countermeasures to protect ASV from voice conversion and speech synthesis attacks. The second edition of ASVspoof held in 2017 switched focus to the mitigation of replay attacks [8, 9, 10], the focus in this paper.

The many submissions to the ASVsoof 2017 challenge can be classified into one of two different approaches. The first set of approaches involves the combination of hand-crafted features with generative classifiers such as Gaussian mixture models (GMM) and i-vectors/PLDA systems, e.g. [11, 12, 13, 14, 15, 16, 17, 18]. The second of approaches explored the use of discriminative classifiers such as support vector machines (SVMs) and deep neural networks (DNNs) [13, 16, 19, 17, 20, 21, 18].

Deep learning techniques, in particular, proved to be especially successful, with five of the top ten performing systems submitted to the ASVspoof 2017 challenge employing some form of automatic feature learning and/or classification[1]. The work in [16] used convolutional neural networks for the automatic learning of features from magnitude spectrograms with a combination of convolutional and recurrent layers in an end-to-end solution.

End-to-end approaches to anti-spoofing have obvious appeal. In more traditional fields of speech processing, such as ASV, there is a substantial body of knowledge that has been acquired over decades of research. This knowledge has been exploited to design extremely effective hand-crafted features. Even in the case of ASV, though, automatic approaches to feature learning are bringing advances in performance [22]. Research in anti-spoofing is comparatively embryonic. The history of benchmarking evaluations spans only three years and the quest for better-performing, hand-crafted features is a staple of current research [1]. That the natural variation in spoofing attacks is so great makes hand-crafted feature design especially difficult. In the absence of an extensive body of background knowledge or proven features, and with the availability of large datasets of spoofed speech, automatic feature learning and end-to-end solutions present an opportunity to fast track progress.

[1] A summary of top submissions is available at http://www.asvspoof.org/slides_ASVspoof2017_Interspeech.pdf

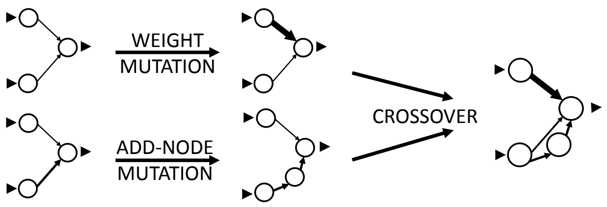


Figure 1: *Mutation of weight (here symbolized by connection thickness), node adding and crossover: the three forms of network evolution. Figure reproduced from [24].*

Motivated by the obvious appeal, by recent work in a similar direction [23] and by the success of the same technique for automatic speaker recognition [24], we have explored a particular approach to automatic feature learning in a truly end-to-end solution to anti-spoofing. It is based upon a class of algorithms known as topology and weight evolving neural networks (TWEANNs), specifically the neuroevolution of augmenting topologies (NEAT) algorithm proposed in [25]. The novel contributions in this paper are four-fold: (i) we present the first application of neuroevolutive learning to the anti-spoofing problem; (ii) we propose a fitness function that is better adapted to audio classification problems; (iii) we demonstrate the merit of automatic learning and end-to-end optimisation in tackling the so-called *generalisation* problem, namely solutions that do not generalise well to test data containing spoofing attacks different to those encountered in training and development, and (iv) we demonstrate that the proposed approach not only improves on generalisation, but that it also brings a significant improvement to the ASVspoof 2017 baseline results.

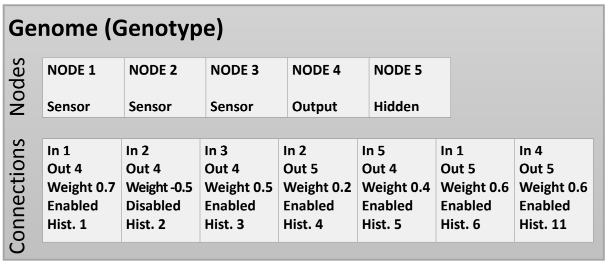
The remainder of the paper is organised as follows. Section 2 provides an overview of the NEAT algorithm and describes recent work that facilitates its application to audio classification tasks. Section 3 introduces a new fitness function tailored to the anti-spoofing problem. Experimental setup and results are the focus of Sections 4 and 5. Conclusions and ideas for future research are presented in Section 6.

2. NEAT

This section introduces the neuroevolution of augmenting topologies (NEAT) algorithm and describes its application to acoustic signals and their classification. Also described is a modification to the fitness function which was found to give better performance when NEAT was applied to audio classification tasks. The focus of this section is on *past* work. New contributions are the focus of Section 3.

2.1. Original work

The NEAT algorithm was introduced by Stanley and Miikkulainen in 2002 [25]. In similar fashion to other topology and weight evolving neural network (TWEANN) approaches, NEAT is a particularly elegant algorithm which exploits the appeal of both genetic algorithms and neural networks. The NEAT algorithm is initialised with a pool of candidate networks, all potential solutions to a given classification task. Inputs may be data samples or features whereas outputs are some form of score. The pool of networks evolves iteratively over many iterations, with the pool of networks within one iteration forming a generation of solutions. At each iteration, networks can mutate through the addition of new nodes or connections, the modification of connection weights or upon a crossing over of a pair of different networks (see Fig. 1).



Genome (Genotype)

Nodes

NODE 1	NODE 2	NODE 3	NODE 4	NODE 5
Sensor	Sensor	Sensor	Output	Hidden

Connections

In 1	In 2	In 3	In 2	In 5	In 1	In 4
Out 4	Out 4	Out 4	Out 5	Out 4	Out 5	Out 5
Weight 0.7	Weight -0.5	Weight 0.5	Weight 0.2	Weight 0.4	Weight 0.6	Weight 0.6
Enabled	Disabled	Enabled	Enabled	Enabled	Enabled	Enabled
Hist. 1	Hist. 2	Hist. 3	Hist. 4	Hist. 5	Hist. 6	Hist. 11

Figure 2: *A NEAT genotype is a direct and self-contained textual representation of an unique network, which contains (as in nature) more information than that which can be observed in the resulting structure. Figure reproduced with permission from [25].*

> In order to avoid confusion, it is stressed that TWEANNs make no use of backpropagation or gradient descent algorithms during training. Networks evolve only as a result of mutation according to the processes illustrated in Fig. 1. A measure of performance is required to control network selection and evolution. Performance is gauged according to the *fitness function*.

Since NEAT operates on a set of minimal initial structures and *augments* complexity gradually at each generation in order to solve the problem in hand, even fully evolved networks tend to be considerably less complex than typical deep neural network solutions. The relatively simple structure of NEAT networks means that they are well suited to embedded applications.

Network characteristics are described in the form of a genotype with *direct encoding*. The genotype is a compact representation of the structure (units and connections) and associated parameters (weights). The information encoded within identifies a unique individual (see Fig. 2). The chronological sequence of structural changes that occur between generations are recorded in the form of *historical markings*. In resolving the so-called structure permutation problem [26] they provide an elegant means of representing gene alignment which dictates which networks among a population are compatible for crossover.

Evolution is controlled according to the concept of so-called *fitness*. The fitness function is used to determine which networks within a current generation will contribute to form the next generation of networks (see Fig. 3). Fitness is evaluated according to the similarity between classification results with the labels from suitable quantities of training data. The NEAT algorithm is hence a form of supervised learning.

Structural innovation is fuelled by protecting a proportion of networks within a population that may not have the highest fitness. These networks may nonetheless have potential to produce better performing networks in later generations. This is achieved by clustering the population of networks into sets of *species* according to a measure of compatibility encoded in the genotype. At every iteration, all networks within the new population are assigned to the species with which they are most compatible. In the event that one or more networks are incompatible with the current set of species, then a new species is created. The best individuals belonging to each species are then

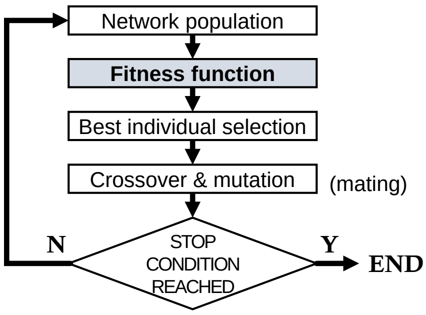


Figure 3: *An illustration of one iteration of evolution: the performance of each network in a population is assessed by the means of a fitness function and the* best *individuals are selected to form a new generation of networks. Figure reproduced from [24].*

selected, meaning networks compete for the best fitness within a niche. This concept of *speciation* serves to protect novelty and diversity within the population which hence has greater potential as a whole to solve the problem in hand.

NEAT has been applied successfully to a multitude of tasks such as bipedal locomotion [27], automated computer game playing [28], as an approach to general acoustic classification [29], audio effect generation [30] and sound event detection [31]. The work in [29] was the first to apply NEAT to the classification of raw audio.

2.2. Application to raw audio

A high-level overview of the framework proposed in [29] by which NEAT can be applied to the processing and classification of raw audio signals is illustrated in Fig. 4. Inputs consist of a bias unit (set to unity) and an input unit which is fed sample-by-sample by a raw audio signal. The latter is propagated through the network at one activation step per sample. There are two output units, a *gate* unit y_r and a *score* unit y_d, each of which produce one output sample per input sample. The network topology between inputs and outputs is of the form illustrated in Fig. 1. It is naturally recurrent in nature; there is no notion of hierarchical layers and no restrictions on links between units. With the exception of score and gate output units which have identity and binary step activation functions respectively, all units have rectified linear activation functions.

Connections can be made freely between *any* pair of units. As a result, evolved networks may contain cyclical unit connections (e.g. units connected to themselves or to other units which influence their input). This classifies NEAT structures as recurrent neural networks.

The product of the gate and score output units is averaged over K samples[2], thereby producing a weighted score y_w:

$$y_w[i] = \frac{\sum_{j=0}^{K-1} y_d[i-j] \times y_r[i-j]}{\sum_{j=0}^{K-1} y_r[i-j]} \tag{1}$$

where i is the sample index and where the gate output y_r is the weight. As proposed in [29], the weighting produced by the gate can be continuous, or may be constrained to a binary weighting $\{0{,}1\}$. While the behaviour of the gate is learned automatically, it will act naturally as a form of attention mechanism [32, 33], i.e. to emphasise the most salient output scores.

2.2.1. Fitness estimation

As in the original work, network performance is measured through a fitness function. The fitness function is key since it is used as a factor in the control of the evolutionary process. The work in [29] used a generic squared-error-based fitness function defined according to:

$$F = 1/\left[1 + \sum_{i=0}^{N-1} (g[i] - y_w[i])^2\right] \tag{2}$$

where g is a ground truth signal of classification labels, *e.g.* 0 and 1. The summation over N reflects the difference between labels and averaged scores, i.e. the inverted error gives a measure of reliability, or fitness.

Our own investigations using the NEAT algorithm for an automatic speaker recognition task [24] showed that the fitness function in Eq. 2 is not sufficiently informative as a means of guiding evolution. Eq. 2 reflects the average proximity of network scores to ground truth labels, rather than *classification* reliability. The latter is often measured with the application-neutral equal error rate (EER). Use of the EER was also found to be sub-optimal; it reflects the reliability of a classifier at a single operating point, *i.e.*, a fixed threshold.

Being independent to a specific operating point, the receiver operating characteristic (ROC) profile is a more informative measure of classifier reliability. ROC profiles may be reduced to a single scalar by measuring the so-called area under the ROC (AUROC) [34]. The AUROC is well tailored to classification as it is proportional to the ability of a classifier to attribute higher scores to positive trials than to negative trials. The work in [24] reports the replacement of Eq. 2 with an AUROC function calculated using the *trapezoid rule* [35].

2.2.2. Mini-batching

Mini-batch training can be used [24] to manage computational effort and to avoid over-fitting. Mini-batching is employed in a similar manner as with the *stochastic gradient descent* algorithm [36] whereby each iteration of training is performed with different batches of data, each a subset of the training partition. The learning of each generation with a different subset of training data promotes network novelty, reduces computation and encourages the evolution of networks that will have better potential to generalise well to unseen data.

The work in [24] defines a pair of mini-batch parameters M_t and M_i. They represent the fraction of available target and impostor data used for each step for the mini-batch training of an automatic speaker verification system. As an example, the setting of both parameters M_t=M_i=100% is equivalent to no mini-batching, with every generation being fed with the full partition of training data. In contrast, the setting of M_t=M_i=50% implies two mini-batches each comprising half of the available target and impostor training data. In this case, the composition of the mini-batches is reset and re-partitioned at random for *every other* generation.

Given the focus of this paper upon anti-spoofing rather than automatic speaker verification, notations M_t and M_i are replaced with M_b (bona fide speech) and M_s (spoofed speech). This notation is adopted throughout the remainder of this paper.

[2] The work in [29] proposed a flexible streaming/segmental or file-based approach were the value of K is adjusted accordingly. All work reported in this paper relates to a file-based processing approach where K is set to the number of samples in a file.

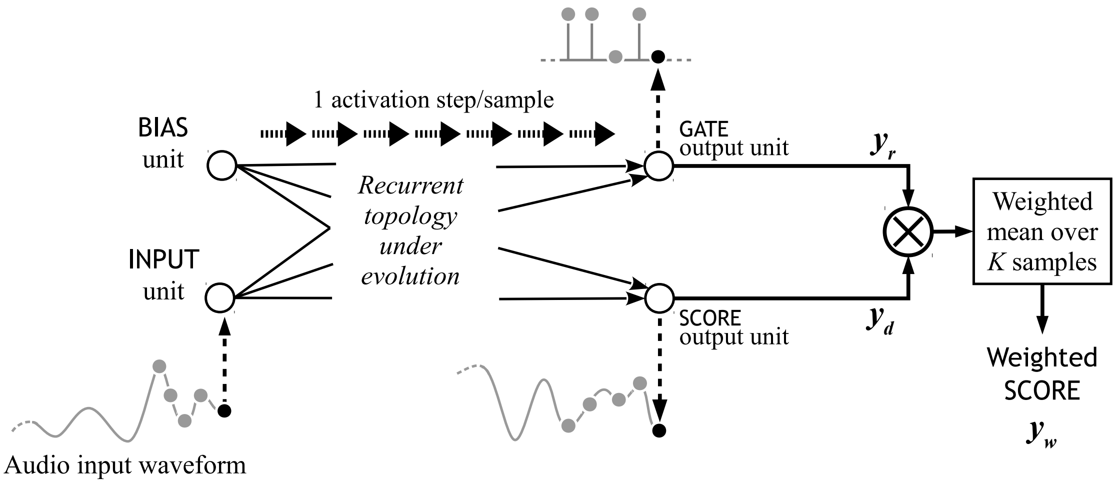


Figure 4: *End-to-end setup and propagation scheme for audio classification. Figure reproduced from [24].*

3. End-to-end anti-spoofing

This section describes the adaptation of the NEAT approach to the anti-spoofing problem. It encompasses the novel contributions claimed in this paper. These comprise a new fitness function which was found to give improved performance in the case of anti-spoofing, in addition to training and optimisation (network selection) procedures.

3.1. Ease of classification

Experiments to evaluate the performance of NEAT for anti-spoofing using the previously reported fitness functions [29, 24] showed a tendency to oscillate around local optima, namely networks in subsequent generations that correct previous classification errors while introducing new ones. Such oscillations can be avoided by using an enhanced fitness function which rewards *progress* rather than raw performance. Progress infers better networks which correct previous classification errors *without* introducing new ones.

An expression for fitness which rewards progress requires the definition of a measure of segment (file) classification ease. Intuitively, this is proportional to how high or how low is the score for segment s compared to the average impostor (spoofed) or target (bona fide) scores respectively; For every network n and **bona fide** segment s with score θ_s, the classification ease is given by:

$$l_{s,n} \leftarrow 1 - \frac{\#\{\textbf{spoofed} \text{ segments with score} > \theta_s\}}{\#\{\textbf{spoofed} \text{ segments}\}} \quad (3)$$

where the right-most term is akin to the false acceptance rate for the given threshold. Conversely, for every **spoofed** segment with score θ_s, the classification ease is given by is given by:

$$l_{s,n} \leftarrow 1 - \frac{\#\{\textbf{bona fide} \text{ segments with score} < \theta_s\}}{\#\{\textbf{bona fide} \text{ segments}\}} \quad (4)$$

where the right-most term is now akin to the false rejection rate for the given threshold. A *pooled* measure of the classification ease may then be obtained by averaging the classification ease over the number G of networks in the population:

$$p_s \leftarrow \frac{\sum_n l_{s,n}}{G} \quad (5)$$

where $l_{s,n}$ is set according to Eqs. 3 or 4 depending on whether segment s is a bona fide or spoofed respectively. A measure of network fitness F is then estimated across all segments accordingly to:

$$F = \frac{\sum_s l_{s,n}(1 - p_s)}{\sum_s (1 - p_s)} \quad (6)$$

where $(1 - p_s)$ acts to weight the contribution of the classification ease for segment s, and network n. This approach to fitness estimation is from here on in referred to as the ease of classification (EOC).

According to Eq. 6, the correct classification of segments that were already correctly classified by networks in an earlier generation thus contributes little to the estimation of fitness for networks in the subsequent generation; there is little reward for learning what was already known by an earlier generation. The EOC approach to fitness estimation steers evolution to classify correctly a diminishing number of *difficult* segments.

3.2. Training

The size of each population is fixed across generations and set to 150 networks. The algorithm is initialised (generation zero) with 150 minimal perceptron-like networks, all of which share the common setup described in Section 2.2. All input signals are normalised to within a range of [-1,1]. The choice of rectified linear unit activation functions results in faster processing, but also increases the chances of saturation. The random initialisation of weights within a $[-4, 4]$ range was found to manage the risk of saturation.

Experiments were conducted with both AUROC (Section 2.2) and EOC (Section 3.1) fitness functions, with and without mini-batching. Audio signals containing either bona fide or spoofed speech are fed to each network segment-by-segment (file-by-file) and the network is trained in order to distinguish between the two. The AUROC fitness function is evaluated with K in Eq. 1 set to the number of samples in each file. All networks are reset after the processing of each file.

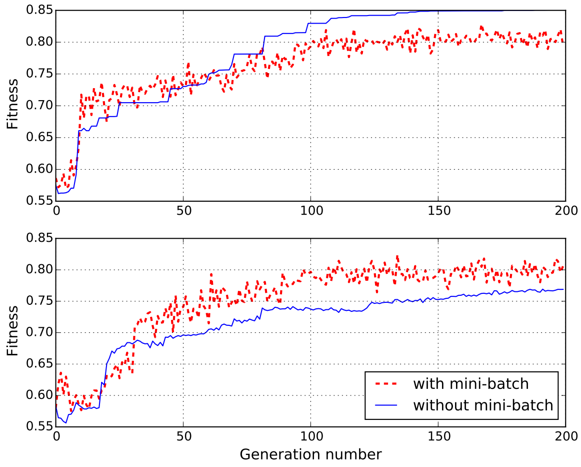


Figure 5: *An illustration of fitness evolution for the fittest network of each generation when using AUROC (top) and EOC (bottom) fitness functions. The dashed red profiles illustrates the fitness evolution with mini-batch training whereas the mostly-monotonic blue profiles shows the fitness without mini-batch training.*

At each iteration (generation), a subset of the fittest (best performing) networks among each species is determined and used to evolve the next generation of networks according to the procedure outlined in Section 2.1. Evolution proceeds either until the fitness converges or until a pre-determined maximum number of generations is reached.

Fig. 5 depicts the improvement in network fitness (vertical axis) over 200 generations (horizontal axis). Illustrated is the evolution in fitness for four different configurations: the two fitness metrics AUROC and EOC, both with and without mini-batch training ($M_b = 25\%$ and $M_s = 33\%$). Each point on each profile shows the fitness of the single, fittest network among the 150 in the population. In both graphs the dashed red curves relate to mini-batch training. Neither profile is monotonic since the data changes at each generation. Conversely, solid blue curves show fitness without mini-batch training ($M_b = M_s = 100\%$), hence the largely monotonic profiles (the fitness of EOC optimised networks is not strictly monotonic on account of the different weights applied to each segment during fitness estimation, as described in Section 3.1).

Profiles in Fig. 5 show that mini-batching is of more benefit when used with the EOC fitness function. Changes in training data can be interpreted as optimisation towards a moving target. This fuels novelty instead of over-fitting to a fixed training set. These observations would suggest a potential for better generalised spoofing detection. It should be noted, however, that the final objective is not higher fitness for training data, but the *classification reliability* assessed using test data.

3.3. Testing

Networks with high measures of fitness may not necessarily be those which give the best performance in terms of the spoofing detection EER. This is especially true when using mini-batch since one random subset of training data could be fortuitously easier than another subset (or indeed the full set). In addition, measures of fitness derived using the EOC fitness function may not be especially well correlated with classification performance; increases in EOC reflect the learning of new information rather than raw performance.

As a result, the fittest network identified from training may not be that which gives the lower EER. In order to observe the evolution in classification performance, the 10 best networks of each generation identified using the fitness function are evaluated using development data and with an EER metric.

The single network with the lowest EER within each group of 10 is named the *generation champion* and the overall lowest EER network among the set of generation champions is denoted the *grand champion*. The latter is selected for testing/evaluation where it is used without further modification.

4. Experimental setup

This section describes the database, protocol and metric used for all experiments reported in this paper. Also described is the baseline system and specific configuration details for the proposed end-to-end approach to anti-spoofing.

4.1. Database, protocol and metric

Experiments were performed using Version 2.0[3] of the ASVspoof 2017 database [37]. The database originates from the RedDots database[4] which was collected by volunteers from across the globe using mobile devices, in the form of smartphones and tablet computers. While the RedDots database was collected to support research in text-dependent automatic speaker verification, the ASVspoof 2017 database was adapted from it in order to support research in anti-spoofing. It contains sets of bona fide (genuine) and replayed speech [38, 39, 9]. In order to simulate replay spoofing attacks, the bona fide partition of the ASVspoof 2017 database was replayed and then recaptured using a variety of different loudspeakers and recording devices in heterogeneous acoustic environments.

The standard protocol relates to a partition of the database into training, development and evaluation subsets, details of which are presented in Table 1. The three subsets are mutually disjoint in terms of speakers and of data collection sites. Experiments reported in this paper were performed with the extended protocol whereby both training and development were performed with pooled training and development partitions (train+dev). The evaluation subset contains data collected using 57 replay configurations, 49 of which differ to those used in the collection of the training and development subsets. Differences in replay detection performance between the training/development and evaluation subsets serve to gauge the generalisation of spoofing countermeasure solutions.

The ASVspoof 2017 evaluations assessed the performance of spoofing countermeasures in isolation to automatic speaker verification. The standard metric is the application-independent equal error rate (EER). It is used for all assessments reported in this paper.

4.2. Baseline

The ASVspoof 2017 Version 2.0 database was released in order to correct data anomalies detected subsequent to the official evaluation. Being released in 2018, the only published results relating to Version 2.0 are those for the official ASVspoof

[3] `http://dx.doi.org/10.7488/ds/2301`

[4] `https://sites.google.com/site/thereddotsproject/`

Table 1: Statistics of the ASVspoof 2017 database version 2.

Subset	# spk	# replay sessions	# replay configs	# utterances bona fide	replay
Training	10	6	3	1507	1507
Devel.	8	10	10	760	950
Eval.	24	161	57	1298	12008
Total	42	177	61	3566	14466

2017 baseline system[5]. It uses a constant Q cepstral coefficient (CQCC) [40, 41] frontend and a traditional Gaussian mixture model (GMM) back-end [42, 43]. Classifier scores are computed as the log-likelihood ratio for the test utterance given bona fide and replayed speech models. This paper considers only the extended protocol baseline for which training and development are performed using pooled training and development dataset (train+dev). Baseline results for the extended protocol are presented to the top of Table 2.

4.3. End-to-end anti-spoofing

The end-to-end algorithm described in this paper was applied to distinguish between bona fide and spoofed speech. All networks are configured according to the common setup described in Section 2.2 and as depicted in Fig. 4. Experiments were conducted with four different configurations comprising AUROC (Section 2.2.1) and EOC (Section 3.1) fitness functions with and without mini-batch training (Section 2.2.2). Configurations in which mini-batch is adopted are labeled with m (see Table 2). Each configuration was run for 500 generations.

When applied, mini-batch training is performed with bona fide speech partitioned into four mini-batches (M_b=25%) of approximately 17 minutes each. Spoofed data is partitioned into three mini-batches (M_s=33%), approximately 21 minutes each. The discrepancy between bona fide and spoofed speech is due to the greater variation in spoofed speech, the reliable modelling of which requires greater quantities of data in each batch.

Once the training of a generation is completed, the performance of networks for that generation is assessed according to the procedure described in Section 3.3. This assessment is performance using pooled training and development partition data (see Section 4.1).

5. Experimental results

This section describes experimental results, starting with an illustration of the evolutionary behaviour of the end-to-end approach to spoofing detection and then an assessment of performance in terms of the EER. Also discussed here is the behaviour of the gate.

5.1. Evolutionary behaviour

An illustration of the evolutionary behaviour of the end-to-end approach to spoofing detection is illustrated in Fig. 6. Two profiles show the evolution in EOC$_m$ (top blue profile) and the number of network node connections (green profile) of the EOC-fittest. The lower magenta profile shows the EER for the champion of each generation (the *generation champions*) estimated using training/development The single network selected for the testing/evaluation is that which produces the lowest EER for the training/development data (orange dot). This network is designated as the *grand champion* network.

Table 2: End-to-end spoofing detection performance for the ASVspoof 2017 V2.0 database and extended protocol.

	Train+Dev	**Eval**
Baseline	0.14%	23.4%
AUROC	20.9%	28.2%
AUROC$_m$	27.4%	24.2%
EOC	20.3%	19.2%
EOC$_m$	18.7%	18.2%

The fitness is seen to increase gradually as the end-to-end approach to anti-spoofing learns to discriminate between bona fide and spoofed speech, gradually increasing network complexity as it proceeds. Improvements in fitness are largely accompanied by decreases in EER. After approximately 350 iterations, the EER seems to converge, with the best performing network being that from the 484th generation and having 198 connections.

5.2. Spoofing detection performance

Results are presented in Table 2 for the baseline systems and the for the end-to-end system with AUROC and EOC fitness functions, with and without mini-batching (denoted by subscript m). Results for the EOC fitness function are either similar to or better than those for the AUROC fitness function. Mini-batching appears to offer inconsistent results for the AUROC fitness function; performance degrades for train+dev. but improves for evaluation. For the EOC fitness function, improvements are consistent across the two sets.

Of particular interest is the stability or generalisation achieved by the end-to-end system. Performance for the baseline system is seen to degrade substantially between the two sets (train+dev and evaluation). In contrast, the best results achieved with the end-to-end approach using the EOC fitness function and mini-batch training is not only substantially better, but also consistent across the two disjoint data sets (18%).

5.3. Gate operation

The gate acts to identify salient information in the network output. This is a form of an attention mechanism. As such, it is of interest to investigate its behaviour. Even so, the gate operates on the *output* stream rather than the *input* stream. Coupled with the recurrent nature of the network which maps inputs to outputs, this impedes a straightforward interpretation of its behaviour; it is difficult to interpret gate behaviour at the output with respect to the acoustic stream at the input.

Our investigations thus far show that the gate generates a somewhat periodic signal during both speech and non-speech intervals. This would indicate that information during both are of use to the detection of replay spoofing attacks. Further analysis would involve a deeper examination of how to link gate behaviour at the output to information at the input. This study is left for future work.

[5] `http://www.asvspoof.org/data2017/baseline_CM.zip`

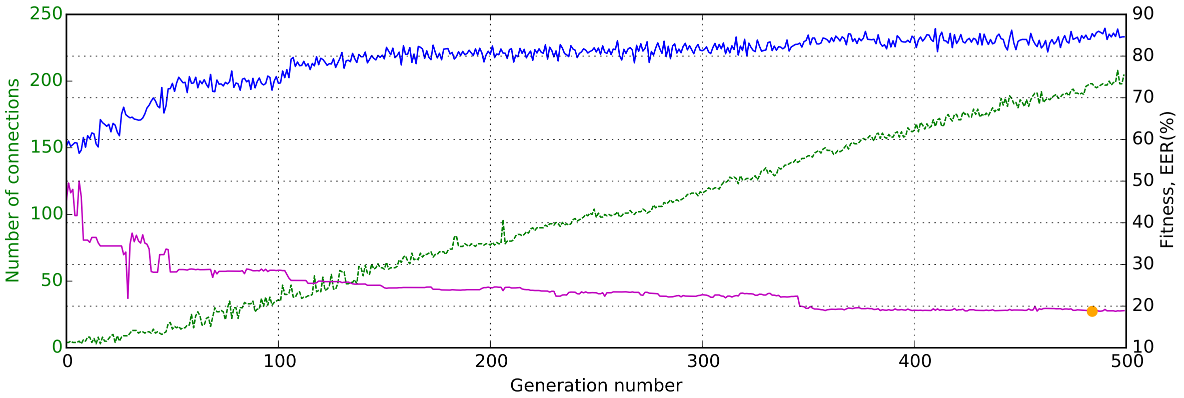


Figure 6: *Evolution of 500 generations with an EOC fitness function with mini-batch training. The upper blue profile shows the EOC-derived fitness of the fittest network in each generation. The highest fitness is obtained in generation 490. The green profile is the complexity (number of connections) in each network. The lower magenta profile is the EER of generation champions estimated using pooled training and development data. It reaches a minimum value in generation 484 (marked by an orange dot). This is the grand champion network that is chosen for testing on the evaluation set.*

6. Conclusions and future work

This paper reports a truly end-to-end approach to the problem of spoofing detection. End-to-end techniques that avoid a reliance upon hand-crafted features are assumed to offer better potential for spoofing detection, and especially generalisation when the cues indicative of spoofing can vary considerably and are largely unpredictable in practice. The paper show how the neuroevolution of augmenting topologies can be applied successfully to this task. Critical to performance is the proposed progress-rewarding fitness function which steers the evolutionary process progressively towards the reliable classification of a diminishing number of difficult trials. Coupled with a mini-batch training procedure, this particular quality of the proposed solution preserves generalisation.

Results for the ASVspoof 2017 Version 2.0 database show improvements to both generalisation and raw performance. Equal error rates for the end-to-end approach represent a 22% relative reduction compared to the baseline system. A particularly appealing feature of the end-to-end approach is the gate, which acts as a form of in-built attention mechanism which serves to distinguish the most reliable information in the network output. This aspect of the end-to-end solution requires further investigation in order to interpret its behaviour with respect to information present in the acoustic input. The findings of such a study, while left for further work, will help to determine precisely what information helps most to differentiate between bona fide and replayed speech.

7. References

[1] T. Kinnunen and H. Li, “An overview of text-independent speaker recognition: From features to supervectors,” *Speech Communication*, vol. 52, no. 1, pp. 12–40, 2010.

[2] J. H. L. Hansen and T. Hasan, “Speaker recognition by machines and humans: a tutorial review,” *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[3] K.A. Lee, B. Ma, and H. Li, “Speaker verification makes its debut in smartphone,” *IEEE signal processing society speech and language technical committee newsletter*, February 2013.

[4] N. Evans, T. Kinnunen, and J. Yamagishi, “Spoofing and countermeasures for automatic speaker verification,” in *Proc. INTERSPEECH*, 2013, pp. 925–929.

[5] R. G. Hautamäki, T. Kinnunen, V. Hautamäki, and A.-M. Laukkanen, “Automatic versus human speaker verification: The case of voice mimicry,” *Speech Communication*, vol. 72, pp. 13–31, 2015.

[6] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi, F. Alegre, and H. Li, “Spoofing and countermeasures for speaker verification: A survey,” *Speech Communication*, vol. 66, pp. 130 – 153, 2015.

[7] Z. Wu, J. Yamagishi, T. Kinnunen, C. Hanilci, M. Sahidullah, A. Sizov, N. Evans, M. Todisco, and H. Delgado, “ASVspoof: the automatic speaker verification spoofing and countermeasures challenge,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 4, pp. 588–604, 2017.

[8] J. Villalba and E. Lleida, “Preventing replay attacks on speaker verification systems,” in *2011 Carnahan Conference on Security Technology*, Oct 2011, pp. 1–8.

[9] F. Alegre, A. Janicki, and N. Evans, “Re-assessing the threat of replay spoofing attacks against automatic speaker verification,” in *International Conference of the Biometrics Special Interest Group (BIOSIG)*, Sept 2014, pp. 1–6.

[10] M. Grzywacz J. Gaka and R. Samborski, “Playback attack detection for text-dependent speaker verification over telephone channels,” *Speech Communication*, vol. 67, pp. 143 – 153, 2015.

[11] R. Font, J. M. Espn, and M. J. Cano, “Experimental analysis of features for replay attack detection results on the ASVspoof 2017 challenge,” in *Proc. INTERSPEECH*, 2017, pp. 7–11.

[12] H. A. Patil, M. R. Kamble, T. B. Patel, and M. H. Soni, “Novel variable length teager energy separation based instantaneous frequency features for replay detection,” in *Proc. INTERSPEECH*, 2017, pp. 12–16.

[13] W. Cai, D. Cai, W. Liu, G. Li, and M. Li, “Countermeasures for automatic speaker verification replay spoofing attack : On data augmentation, feature representation,

classification and fusion," in *Proc. INTERSPEECH*, 2017, pp. 17–21.

[14] S. Jelil, R. K. Das, S. R. M. Prasanna, and R. Sinha, "Spoof detection using source, instantaneous frequency and cepstral features," in *Proc. INTERSPEECH*, 2017, pp. 22–26.

[15] M. Witkowski, S. Kacprzak, P. elasko, K. Kowalczyk, and J. Gaka, "Audio replay attack detection using high-frequency features," in *Proc. INTERSPEECH*, 2017, pp. 27–31.

[16] G. Lavrentyeva, S. Novoselov, E. Malykh, A. Kozlov, O. Kudashev, and V. Shchemelinin, "Audio replay attack detection with deep learning frameworks," in *Proc. INTERSPEECH*, 2017, pp. 82–86.

[17] L. Li, Y. Chen, D. Wang, and T. F. Zheng, "A study on replay attack and anti-spoofing for automatic speaker verification," in *Proc. INTERSPEECH*, 2017, pp. 92–96.

[18] K. R. Alluri, S. Achanta, S. R. Kadiri, S. V. Gangashetty, and A. K. Vuppala, "SFF anti-spoofer: IIIT-H submission for automatic speaker verification spoofing and countermeasures challenge 2017," in *Proc. INTERSPEECH*, 2017, pp. 107–111.

[19] X. Wang, Y. Xiao, and X. Zhu, "Feature selection based on cqccs for automatic speaker verification spoofing," in *Proc. INTERSPEECH*, 2017, pp. 32–36.

[20] Z. Chen, Z. Xie, W. Zhang, and X. Xu, "Resnet and model fusion for automatic spoofing detection," in *Proc. INTERSPEECH*, 2017, pp. 102–106.

[21] P. Nagarsheth, E. Khoury, K. Patil, and M. Garland, "Replay attack detection using dnn for channel discrimination," in *Proc. INTERSPEECH*, 2017, pp. 97–101.

[22] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, "Deep Speaker Feature Learning for Text-independent Speaker Verification," *arXiv:1705.03670 [cs]*, May 2017.

[23] H. Muckenhirn, M. Magimai-Doss, and S. Marcel, "End-to-End Convolutional Neural Network-based Voice Presentation Attack Detection," in *IEEE IAPR International Joint Conference on Biometrics (IJCB)*, 2017.

[24] G. Valenti, A. Daniel, and N. Evans, "End-to-end automatic speaker verification with evolving recurrent neural networks," in *Speaker Odyssey 2018*.

[25] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary computation*, vol. 10, no. 2, pp. 99–127, 2002.

[26] N. J. Radcliffe, "Genetic set recombination and its application to neural network topology optimisation," in *Neural Computing and Applications*, 1993, number 1, pp. 67–90.

[27] B. Allen and P. Faloutsos, "Complex networks of simple neurons for bipedal locomotion," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. 2009, pp. 4457–4462, IEEE.

[28] M. Hausknecht, J. Lehman, R. Miikkulainen, and P. Stone, "A neuroevolution approach to general atari game playing," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 6, no. 4, pp. 355–366, 2014.

[29] A. Daniel, "Evolving recurrent neural networks that process and classify raw audio in a streaming fashion," in *Proc. INTERSPEECH*, 2017.

[30] I. Jordal, "Evolving artificial neural networks for cross-adaptive audio effects," M.S. thesis, NTNU, 2017.

[31] C. Kroos and M. Plumbley, "Neuroevolution for sound event detection in real life audio: A pilot study," *Detection and Classification of Acoustic Scenes and Events (DCASE 2017) Proceedings*, 2017.

[32] C. Olah and S. Carter, "Attention and augmented recurrent neural networks," in *Distill*, 2016.

[33] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *arXiv preprint arXiv:1508.01211*, 2015.

[34] T. Fawcett, "An introduction to ROC analysis," in *Pattern Recognition Letters*, 2006, number 27, pp. 861–874.

[35] J. A. C. Weideman, "Numerical integration of periodic functions: a few examples," in *The American Mathematical Monthly*, 2002, number 109, pp. 21–36.

[36] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

[37] T. Kinnunen, M. Sahidullah., H. Delgado, M. Todisco, N. Evans, J. Yamagishi, and K. Aik Lee, "The ASVspoof 2017 challenge: Assessing the limits of replay spoofing attack detection," in *Proc. of INTERSPEECH*, 2017, pp. 2–6.

[38] J. Villalba and E. Lleida, *Biometrics and ID Management: COST 2011 European Workshop, BioID 2011, Brandenburg (Havel), Germany, March 8-10, 2011. Proceedings*, chapter Detecting Replay Attacks from Far-Field Recordings on Speaker Verification Systems, pp. 274–285, 2011.

[39] Z. Wu, S. Gao, E.S. Chng, and H. Li, "A study on replay attack and anti-spoofing for text-dependent speaker verification," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA*, 2014, pp. 1–5.

[40] M. Todisco, H. Delgado, and N. Evans, "A new feature for automatic speaker verification anti-spoofing: Constant Q cepstral coefficients," in *Proc. Odysssey*, Bilbao, Spain, 2016, pp. 283–290.

[41] M. Todisco, H. Delgado, and N. Evans, "Constant Q cepstral coefficients: A spoofing countermeasure for automatic speaker verification," *Computer Speech & Language*, vol. 45, pp. 516 – 535, 2017.

[42] D.A. Reynolds, T.F. Quatieri, and R.B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, 2000.

[43] Douglas A. Reynolds, "Gaussian mixture models," in *Encyclopedia of Biometrics*, 2009.