Intelligent speech handover for smart speakers through deep learning: a custom loss function approach

Vijaya Nirmala Mitnala, Martin J. Reed, John Bicknell, and Joyraj Chakraborty

***Abstract*—The consistent growth of the smart speaker market has established far-field speech communication as an alternative to traditional handsets. When multiple smart speakers are used, a mechanism for seamless handover is needed, which is not currently supported. This paper presents two novel contributions that, together, enable seamless handover: using speech signals to select a suitable smart speaker through machine learning; and, reduction in media disruption during handover by local modifications to the session initiation protocol (SIP). The proposed solution uses prediction based on a one-dimensional convolutional neural network (1DCNN) and a custom loss function. A comprehensive evaluation with multiple datasets incorporating different types of audio signals, movement loci, and, varying room scenarios demonstrates the effectiveness of the suggested method in predicting the most appropriate smart speaker. Our proposal is shown to be highly effective when compared against a previously proposed predictor or using a standard 1DCNN loss function and operates with a low computational cost, suitable for consumer smart speakers.**

***Index Terms*—Smart speakers, Seemless speech handover, Smart speech mobility, Session Initiation Protocol, Convolutional Neural Networks.**

I. Introduction

SINCE 2015 there has been rapid and sustained growth in the number of connected devices around the home; by 2025 it is expected to reach over 20 billion globally [1]. Many of these *next-gen* consumer electronics - such as smart speakers, laptops, tablets, smartphones, and electronic portals - are able to support video and/or audio communication sessions [2]. Whilst this has led to much greater flexibility in how and where we communicate, little effort has been made to unify or simplify the consumer experience when it is across multiple devices and multiple environments.

Until recently, such unification was difficult to foresee and today, many of the services related to smart devices remain in siloed ecosystems. However, there are promising signs of the industry moving towards open connectivity standards with the adoption of Matter [3]. Indeed, this is also foreseen in the research literature as the *next-gen* consumer electronics [2]. Next-gen consumer electronics will open up new opportunities for device manufacturers and telecommunications providers alike to deliver features that enhance consumer experience without being bound to a single brand or service silo [4]. This paper considers smart mobility in the home for a consumer speech application delivered through smart speakers.

One notable research gap lies in achieving seamless handover between two or more smart speakers during a voice call. While existing studies have explored talker location [5] and vertical handover [6] between different networked systems, there is a lack of research combining these concepts for consumer applications involving multiple smart speakers within a single environment (room), known as *horizontal handover*. Additionally, existing SIP research has not addressed seamless handover. Today, if a user wishes to change smart speakers during a call, it typically involves manually ending the session and starting a new session on the new target smart speaker. The key objectives of seamless speech handover are to (a) always select the smart speaker that will deliver the best possible audio quality for the user and their environment, and (b) to provide a universally adoptable method for managing session handover between standard session initiation protocol (SIP) clients.

This paper proposes a novel solution for the voice call handover between smart speakers within a single audio environment, such as a domestic living room as shown in Fig. 1. Here we can see a user initially using smart speaker 1, for a voice call to a remote corresponding node (CN) and then moving to smart speaker 2 in the same room (or maybe a neighbouring room). It is advantageous to move the call seamlessly between smart speakers 1 and 2, but this is not currently possible as existing SIP mechanisms exhibit significant latency during device handover [6] and current location systems within a room require calibration that is not suitable for consumers. For example, while there have been attempts at determining user location in a room, these have often relied either upon tracking systems (e.g. video capture) [7] or on audio detection that makes use of accurate device and audio environment calibration [5]. Neither approach is well-suited to practical smart speaker usage; devices are normally placed for convenience rather than optimal acoustic performance, devices may be moved to different locations in the room and the auditory environment of the room may change when moving furniture or due to the presence and movement of other people in the room. To avoid this calibration dependency, our research proposes a novel method for identifying the suitable smart speaker using multivariate audio signal features. This approach utilises deep learning techniques, particularly one-dimensional neural network (1DCNN) [8], leveraging its effectiveness

Dr. Vijaya Nirmala Mitnala, and Professor Martin J. Reed, are with the Department of Computer Science and Engineering, University of Essex, Colchester, UK (Email: vm20821@essex.ac.uk, mjreed@essex.ac.uk).

John Bicknell is with British Telecommunication Plc, Ipswich, UK (Email: john.bicknell@bt.com).

Dr. Joyraj Chakraborty is with the Department of Engineering Science, University of Oxford, UK (Email: joyraj.chakraborty@eng.ox.ac.uk).

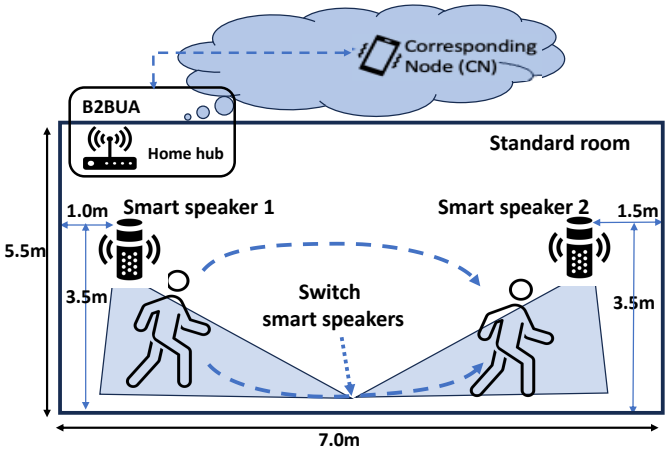


Fig. 1: Example smart speech mobility consumer application with two smart speakers in a room (height of 2.4m not shown)

in analysing time-series multivariate audio signal data. Our motivation for using such machine learning is that we want to train a system that works solely on the one-dimensional audio data obtained from the smart speaker microphones without relying upon other data such as two or three-dimensional positioning information. A deep convolution neural network has the advantage that it can be trained on a suitably large dataset, which will create filters that are thus designed to solve the target problem. Later we show that our solution can run with relatively low complexity by using suitable audio features that are averaged over the time-frame of movement rather than each audio sample.

In summary, the novel contributions encompass:

- the detection of a suitable smart speaker using a 1DCNN based upon only the audio signal features without calibration or knowledge of the location of the smart speakers;
- a custom loss function in the 1DCNN training that is problem-specific to improve over standard machine learning (ML) loss functions;
- and, alternative SIP signalling approaches for mid-call personal mobility providing seamless session handover.

These contributions are shown to require very low computational effort to allow deployment in contemporary consumer smart speakers.

In Section II, we review related literature, followed by the presentation of the problem statement in Section III. Subsequently, we present our proposed methods for both smart speaker detection and SIP in section IV. The advantages of our proposal are demonstrated through the results presented in Section V. We provide discussion on the proposed approach in Section VI before drawing conclusions in Section VII.

II. Related Work

A. Smart speaker technology

With increasing consumer drives for connectivity, the need for smart homes equipped with smart speakers has become more pronounced than ever with an estimated 200 million smart speakers purchased by 2022 and an estimated growth to a $ 21 billion market by 2027 [9]. Smart speakers serve as central hubs that streamline various aspects of daily life, offering convenience, efficiency, and enhanced control over home environments. Voice-activated speakers with virtual assistants can perform tasks and provide information through voice commands. Key components include advanced speech recognition, high-quality audio for music and responses, and beamforming microphones to capture commands effectively.

Localising talkers using speech is a key feature of smart speaker technology for speech mobility. Sound source localisation (SSL) is one way to determine the talker's location and has been widely studied for applications like smart home systems, video conferencing, robotics, speech enhancement, augmented reality, and surveillance systems [10]–[16]. The authors in [17], [18] used traditional signal processing techniques, such as voice power or the time-difference of arrival (TDOA) method, for this task. These methods require prior knowledge of the acoustic environment, such as room dimensions and wall surface area, *etc*.

There have been recent proposals to adopt learning-based methods for sound localisation, eliminating the need for traditional measurements of distance [5], [19]–[21]. Within this evolving paradigm, audio attributes such as magnitude squared coherence (MSC), signal magnitude (A), direct-to-reverberant ratio (DRR), and mel-frequency cepstral coefficients (MFCC) [22] serve as determining features, playing a pivotal role in advancing feature-based machine learning approaches for superior sound localisation and recognition. Baimirov et al. [23] discuss research advancements in smart speaker technology, including the use of microphone signals with CNN techniques and time-series filters like Kalman filters for speech recognition and talker location; but unlike our work, they do not find research that detects the suitable smart speaker for a speech session. The research in [19] compares different acoustic source distance parameters and concludes the MSC method is most effective for distance estimation. The studies in [19], [24] discuss using the MSC to compute the DRR for distance estimation. The research in [5] introduces a Gaussian mixture model (GMM) using MSC features from a binaural input to estimate arrival distance. While these research proposals provide compelling reasons for using MSC in distance estimation after calibration in static environments, they do not address the scenario of generic device switching without calibration. An earlier study [25] introduced a method that leveraged MSC along with smoothing filters to enable smart speaker selection without the need for calibration. However, this method was not applicable to the general case of arbitrary movement and varying room/device locations considered in this paper.

Over the last decade, deep learning has significantly improved in this field. Convolutional neural networks (CNN), one of the dominating paradigms in deep learning [26], have exerted a substantial influence on research across diverse domains, such as human activity recognition [27], [28], speech recognition [29], image classification [30], and, various time-series prediction applications [31]–[33] *etc.* In particular, the 1DCNN [8] has achieved good results in processing time-series multivariate audio signal data. This previous work motivates our paper in utilising a 1DCNN along with suitable audio

features to provide robust suitable smart speaker detection for a smart mobility speech application. Deep learning can require significant computation resources and consequently, authors look for green and sustainable solutions such as offloading to/from end-devices [34] or other green cloud solutions for multimedia [35]. In this paper, we take an alternative approach by implementing deep learning such that it only requires a small, negligible, fraction of a smart speaker's processing resource.

B. SIP Personal mobility

The smart speech mobility solution in this research considers session initiation protocol (SIP) for session handling. SIP [36], [37] is widely accepted and heavily used as an application layer signalling protocol in voice over Internet protocol (VoIP) and Internet protocol multimedia sub-systems (IMS) [38]. VoIP/IMS systems are generally implemented as virtualized software systems [39]; consequently, end system user components are easily implemented in home gateways [40] or in a smart speaker as a SIP server. The SIP server can function fundamentally in two modes: the back-to-back user agent (B2BUA) mode and the proxy mode [41]. The B2BUA-based SIP server fully controls and manages the SIP sessions over all the call stages. This study uses a B2BUA-based SIP server because of its strong features among proxy-based SIP servers and because it is the most commonly used scenario. We note that in our system only the single end user is connected to the B2BUA, so SIP load scheduling is not within the scope of this paper as only a single, or small group of callers, will be part of the call at any one time i.e. the B2BUA has a very low signalling load consisting of simply start of a call, end of a call and movements within the consumer location which happens in the order of seconds. However, a network operator, that is coordinating a large number of calls at any one time, will require load scheduling within its central VoIP architecture; two alternative solutions for this load scheduling are presented by Azhari et al. [42] and by Yavis et al [43].

Personal mobility requires a SIP architecture that has the flexibility to dynamically change according to a user's needs and locations. There are a number of approaches to achieve this, one is to use software-defined networking as proposed by Gandotra and Perigo with SDVoIP [44] and this is a useful technique in cases where mobility spans diverse locations. In this work, we can avoid the need for such network-centric mechanisms as our solution is only concerned within the domain of a single consumer, i.e. with a number of smart speakers within a single location that are typically connected via wireless networking such as WiFi. Consequently, our solution is agnostic to the wider SIP implementation although a software-defined solution such as SDVoIP might bring wider benefits to the general SIP architecture of an operator that is outside of the domain of the consumer. Instead, our solution makes use of the simple approach of a single B2BUA which already exists in most consumer end-points.

SIP *personal mobility* [45], [46] allows registering multiple devices to one SIP address for session handling. A call session can be initiated on multiple devices with the same SIP address using the existing SIP support of either parallel or sequential call forking. This facility can be used in dynamic session handover between smart speakers. The use of SIP *personal mobility* has been studied in the past for various smart home applications [18], [47]. This paper uses the SIP *personal mobility* feature for session handling and introduces a novel solution for seamlessly handling sessions during smart speaker handovers.

III. Problem statement

Given N smart speakers $D = d_1 \dots d_n$ in a room with a user at position $p(t)$ at time t using one of the smart speakers to communicate with a remote user at a corresponding node (CN), the aim of this work is to determine an optimal $d_i \in D$ that will provide the highest quality communications experience. For simplicity, we assume that the user is moving and that the smart speakers are static at locations denoted by the smart speaker variable d_i. On first inspection, this appears to be a simple classification problem. However, this simplistic approach leads to several issues. While there may be a number of ways to determine the optimum smart speaker, for the moment we will assume it is the smart speaker d_i that is nearest, according to Euclidean distance, to the user's position $p(t)$ and denoted as $|p(t), d_i|_2$. Later we will comment on this assumption. If we now consider the selection of the optimum smart speaker $\hat{d}_i$ as

$$\hat{d}_i = \arg\min_{d_i \in D} |p(t), d_i|_2 \tag{1}$$

the classification problem in this form thus depends upon determining the location of the user relative to the speakers to obtain $|p(t), d_i|_2$. However, as noted above in I and II, location tracking in most home environments where smart speakers will be used is very difficult. Existing localisation methods use offline measurements to calibrate the localisation system [5], [19]. However, given that users tend to move smart speakers or furniture, arbitrarily, any such calibration would be invalid, resulting in incorrect localisation. Consequently, determining $|p(t), d_i|_2$ itself is not practical and instead, we propose using features from captured audio on each of the smart speakers to estimate the solution to (1). We describe the features we investigated for this work in IV-A2.

In addition to the problem of location estimation, the classification formulation in (1) leads to issues when determining the error (loss function) in a given position. A classical classification problem would evaluate "wrongness" as simple counts on the number of classification errors, but this misses some subtlety in the problem. Consider, for simplicity of description but without loss of generality, a two-device system $D = d_1, d_2$ as shown in Fig. 2. Here we show a talker moving between position p_α to p_Ω. Ideally, the speech should switch the active smart speaker from d_1 to d_2 as the talker moves through position p_2 where the talker's position is equidistant from the two smart speakers. It would be considered an error to switch the active smart speaker at positions p_1 or p_3. However, if we use a simple, binary, classification value to represent this error this clearly does not adequately represent the problem. For example, switching to device d_2 at p_1 would be disturbing as

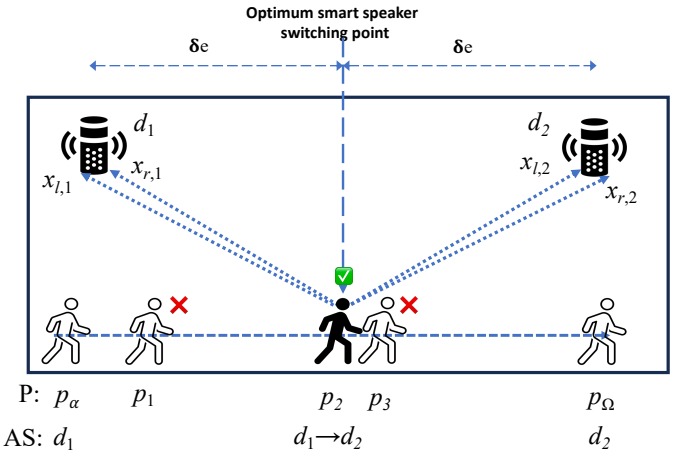


Fig. 2: Example handover process to select an active smart speaker (AS) at a user position (P), with ideal (p_2) and non-ideal (p_1, p_3) handover positions for a talker walking from p_α to p_Ω. Ideally, handover should occur at p_2 and for non-ideal handover, p_1 is perceptually much worse than switching at $p3$.

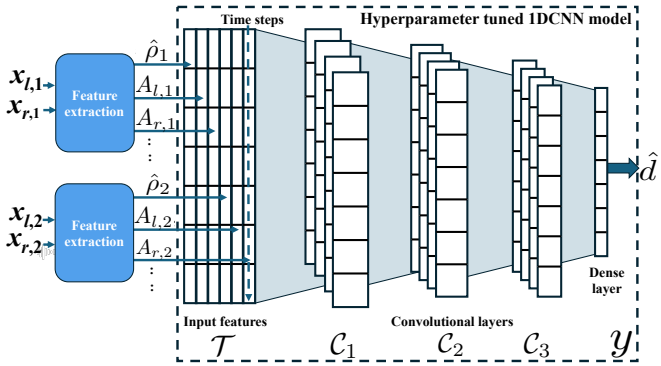


Fig. 3: Complete smart speaker detection pipeline, incorporating feature extraction according to equations (3), (6), and detecting smart speaker $\hat{d}$ using the 1DCNN decision process described in Algorithm 1.

the user is very close to d_1[1]. However, making the same error at p_3 may not even be noticeable to the user as the switching error occurs very close to the ideal position, p_2. Consequently, representing the system as a pure classification value does not adequately represent the problem. The next section presents our solution to this issue using an alternative representation of the error by introducing a custom loss function that takes this into account to correctly train the system.

IV. Proposed Method for Intelligent Smart Speaker Handover

This section describes the proposal for intelligent smart speaker handover in two parts, firstly how nearest smart speaker detection in a home is determined and secondly how SIP can trigger media handover. An overall system diagram is shown in Fig. 3 which shows how the signals from the smart speakers, $x_{l,1} \ldots x_{r,2}$ are used to create features, defined later in IV-A2, which are then fed into a 1DCNN machine learning model as shown in Algorithm 1. Section IV-A will first describe how the classification problem can be modelled as a *custom error* function, that will later be used as a *custom loss* function in machine learning. It then proposes the candidate features that will be extracted for the 1DCNN and describes the structure of this model. Having proposed the nearest smart speaker detection, Section IV-B gives a brief overview of the proposed media handover using SIP.

[1]we are assuming they had stopped at p_1 and the corresponding node is then wrongly switched to talk from d_2.

Input: Time series feature tensor
$\mathcal{T} = [\hat{\rho}_{i1}(t), \hat{\rho}_{i2}(t), A_{l,1}(t), \ldots, A_{r,2}(t)]$
Output: Output $\hat{d}$
Function `Decision(`$\mathcal{T}$`)`**:**
  Data: Filter weights and biases for each layer
  Result: $\hat{d}$
  $w_1 \leftarrow$ Filter weights for layer 1
  $b_1 \leftarrow$ Bias term for layer 1
  $h_1 \leftarrow f(w_1 * \mathcal{T} + b_1)$
  `// where` f `is the relu activation function`
  $w_2 \leftarrow$ Filter weights for layer 2
  $b_2 \leftarrow$ Bias term for layer 2
  $h_2 \leftarrow f(w_2 * h_1 + b_2)$
  $w_3 \leftarrow$ Filter weights for layer 3
  $b_3 \leftarrow$ Bias term for layer 3
  $h_3 \leftarrow f(w_3 * h_2 + b_3)$
  $W \leftarrow$ Weight matrix for the dense layer
  $B \leftarrow$ Bias vector for the dense layer
  $y \leftarrow g(W * h_3 + B)$
  `// where` g `is the softmax activation function`
  $$\hat{d} = \begin{cases} 2 & \text{if } y > 0.5 \\ 1 & \text{if } y \leq 0.5 \end{cases}$$
  return $\hat{d}$

Algorithm 1: The decision process for selecting the optimal smart speaker using the 1DCNN model.

A. Nearest smart speaker detection

A key challenge is to define a prediction function and an associated error function (later to be used as a loss function). To obviate the problem found when using a binary classifier, as described above in III, an alternative could be to consider it as a regression problem which determines the minimum distance to a smart speaker and uses this as a basis for switching. Now in our simple example, again in Fig. 2, the regression error would be significantly smaller if the handover occurred at position p_3 than at p_1. However, while this regression problem has been widely used for systems that attempt to determine exact position, this is not practical in most smart speaker environments as we and others have found that predicting distance in arbitrary rooms with uncalibrated speaker positions and orientations is very difficult [19]. Instead, we choose to consider this a hybrid classification/regression problem by using a normalised regression metric that is then fed to a simple decision classifier based on the output of the normalised

regression model. This hybrid solution is described as an error function that will be used as the custom loss function within the 1DCNN.

1) Custom error/loss function: The custom error/loss function is based on a combination of:

- the normalised distance between the smart speakers;
- a decision variable that indicates whether the ideal smart speaker was used;

The custom error/loss function makes use of ground-truth variables that are known during training (and for evaluation), but are not known by the model during the prediction (running) phase. The proposed formulation of the custom loss function is:

$$L(t) = \left| \frac{|p(t), d_1|_2 - |p(t), d_2|_2}{|d_1, d_2|_2} \right| \delta(p(t), d_1, d_2) \qquad (2)$$

where $\delta(p(t), d_1, d_2)$ is a binary decision variable which is unity if an incorrect smart speaker is selected and zero otherwise. All of the variables in (2) are unknown to a running system as we are assuming that the smart speakers are placed (and possibly moved) by users without any calibration. Consequently, in the 1DCNN, this loss function has to be implemented as a custom function such that the function and corresponding input variables are used during training but then are not used (unknown) during the prediction (running) phase.

2) Feature extraction: In this section, we present a method for intelligently switching between smart speakers within a unified audio environment, using the audio features extracted at each talker's position and at a certain time interval. The inputs to the feature extraction block, shown in Fig. 3, are the audio, $x_{l,1}, x_{r,1}, x_{l,2}, x_{r,2} \ldots$ captured by the smart speakers as shown in Fig. 2. This may include the use of D smart speakers, such as smart speakers equipped with microphones, placed according to user preferences within a room or meeting area. The utilisation of microphone arrays integrated within the smart speaker utilises typical capabilities of such devices [48]. The situation assumes a continuous voice call, and we also presume that the user's movements are relatively gradual compared to the distances between the smart speakers. For instance, the user does not move rapidly from one end of a room to another (running) but rather involves expected movements during a call such as walking. The process for computing the audio features is outlined below. Without loss of generality, we will consider the case where each smart speaker has a microphone array with two microphones as a *pair* denoted l, r. The initial processing system is thus described as:

- Capture an audio sample block, $(u_{l,1}, u_{r,1}), (u_{l,2}, u_{r,2}), \ldots$ separately from the microphone pair (l, r) from each smart speaker $d_i \in D$ at each time step
- Apply A-weighting filter A to each audio signal to emphasise speech over other background noise
- Apply a Hanning window, Han, on each block of audio, using overlap-add to achieve continuity in the signal, $x = \mathrm{Han}(A(u))$
- Calculate a relevant feature (below) from the audio signals $x_{l,i}$, $x_{r,i}$ for D smart speakers, $i \in D$.

These signals, $x_{l,i}, x_{r,i}, \ldots$, are then used to generate features for the machine learning as shown in Fig. 3. A number of candidate features were considered in this work as specified below, these include the magnitude squared coherence, signal magnitude, Mel cepstral coefficients, and Mel magnitude spectrum; each of these is formally defined below.

Magnitude Squared Coherence: The MSC is computed using Welch's cross-power spectral density for the two microphone signals from smart speaker $i \in D$, $x_{l,i}(t)$ $x_{r,i}(t)$, for the left, l, and right, r, channels respectively. Specifically the MSC $\rho_i(t)$ for a block at time t is calculated using the Fourier transform $\mathcal{F}$ of x, $X = \mathcal{F}(x)$:

$$\rho_i(t) = \frac{\left| \hat{P}(X_{l,i}(f,t), X_{r,i}(f,t)) \right|^2}{\hat{P}(X_{l,i}(f,t), X_{l,i}(f,t)) \hat{P}(X_{r,i}(f,t), X_{r,i}(f,t))} \qquad (3)$$

where the cross power spectral density $\hat{P}(X_a(f,t), X_b(f,t))$ is calculated across an N block Fourier transform of $X(f,t)$ from $x(t)$ as:

$$\hat{P}(X_a(f,t), X_b(f,t)) = \frac{1}{N} \sum_{f=0}^{N-1} |X_a^*(f,t) X_b(f,t)| \qquad (4)$$

which denotes $X^*(f,t)$ as the complex conjugate of $X(f,t)$.

Finally the averaged MSC, $\hat{\rho}(t)$ at time, t, across B blocks of size N with sample rate, r is

$$\hat{\rho}(t) = \frac{1}{B} \sum_{k=0, \tau=t+kN/r}^{k=B-1} \rho_k(\tau) \qquad (5)$$

as noted above, this coherence is calculated separately for each smart speaker to give $\hat{\rho}_1$ and $\hat{\rho}_2$ if two smart speakers are used.

Signal magnitude: The absolute signal magnitude A_i, of smart speaker i is calculated from the power spectral density as:

$$A_i(t) = \frac{1}{N} \sum_{f=0}^{N-1} |X^*(f,t) X(f,t)| \qquad (6)$$

This magnitude is calculated for each smart speaker and for the left, l, and right, r, channels as $A_{l,1}$, $A_{r,1}$, and $A_{l,2}$, $A_{r,2}$ if two smart speakers are used.

Mel magnitude spectrum: The Mel magnitude spectrum [22] $m(f_m, t)$:

$$m(f_m, t) = M(|X(f,t)|) \qquad (7)$$

is a warped magnitude spectrum of $|X|$, the discrete Fourier transform of x of a block at time t. Specifically, M is a filterbank that calculates the (mostly) logarithmically spaced frequency magnitude of the signal x at frequencies f_m. Here we used the spacing suggested by Slaney [49] which with a sampling frequency of 16 kHz gives the number of filters $H = 11$ in the filterbank M.

Mel Frequency Cepstral Coefficients: The set of Mel frequency cepstral coefficients (MFCCs) [22] is an alternative *pseudo-time* domain representation of the Mel spaced magnitude spectrum. This is a common representation used in

speech processing and is defined as a set of coefficients, $g_i(n)$, $n = 1 \dots N-2$:

$$g_i(n) = \sqrt{\frac{2}{N}} \sum_{k=1}^{N-2} \log(m(k,n)) \cos\left(\frac{\pi k(2N+1)}{2N}\right) \quad (8)$$

which represents the discrete cosine transform of the logarithmically scaled Mel magnitude spectrum; note n=0, the DC component, and n=N-1 are ignored as they are usually zero magnitude components at these Mel frequencies in audio signals.

3) Machine learning architecture: This research proposes a 1DCNN to identify the most suitable smart speaker. The network processes time-series audio signal data using the multivariate features that were described in Section IV-A2. An overview of the 1DCNN within the system is shown in Fig. 3. In our model, we leverage multiple 1D convolutional layers to identify local patterns and features through the application of convolutional filters (kernels) to the input data. Furthermore, we incorporate the Rectified Linear Unit (ReLU) activation function to infuse non-linear characteristics into the model. The ReLU activation function has the advantage of introducing some non-linearity while maintaining a linear function for a portion of the signal region. We performed a hyperparameter tuning process using the Random Search library from the Keras Tuner library, to identify the number of convolutional layers, filters, and units to achieve optimal performance[2]. The specific hyperparameters that were tuned include the number of filters and the kernel size. The tuning was carried out while using the custom loss function defined in (2).

The input to the 1DCNN layer is a tensor $\mathcal{T}$, where each column $\mathbf{z}(t)$ is a time series vector representing the features described in IV-A2. The full 1DCCN is given in Algorithm 1 and each step is further described below. The input $\mathbf{z}(t)$ is convolved with a kernel $\mathbf{w}(t)$ of size l to obtain the output $\mathbf{C}(t)$, which is described using:

$$\mathbf{C}(t) = \mathbf{z}(t) * \mathbf{w}(t) = \sum_{k=-l}^{l} \mathbf{z}(k) \cdot \mathbf{w}(t-k) \quad (9)$$

The weights of the kernel $\mathbf{w}(t)$ are initialized using He normal initialization [50]. Then, the output of the CNN layer can be represented as:

$$C_i^l = b_i^l + \sum_k \mathbf{C}_k^{l-1} * \mathbf{w}_k^l \quad (10)$$

where C_i^l is the i^{th} output feature at the l^{th} layer, C_k^{l-1} is the k^{th} input feature at the $(l-1)^{th}$ layer, $\mathbf{w}_k$ denotes the convolution kernel at the k^{th} index, and b_i^l is the bias term for the i^{th} output feature at the l^{th} layer.

ReLU activation is applied to the convolution output:

$$ReLU(C_i^l) = \begin{cases} C_i^l & \text{if } C_i^l > 0 \\ 0 & \text{if } C_i^l <= 0 \end{cases} \quad (11)$$

[2]Hyperparameter tuning is a standard approach to optimising model parameters by automatically testing parameters over a reasonable range until good values are reached.

The output of the final dense layer is a single regression value which is then mapped to a class through a binary decision variable, i.e. the choice of smart speaker. The choice of the structure was driven by the observation that smoothing a single feature (e.g. MSC) can aid a very simple decision algorithm. Thus, similarly, the use of 1DCNN layers can be seen to act as time-domain variant trained filters that learn to appropriately process the input data to achieve improved classification performance. The size of the kernels is such that a reasonable history is used to feed into the classification decision, see V-B for the values selected. We found, through experimentation and hyperparameter tuning, that it was best to maintain the time-domain structure throughout the CNN structure, with decreasing kernel size, until the end where the dense layer then reduced the dimensionality to one.

4) Training environment: The training dataset comprises information such as the talker's position from the smart speakers and the signal features, extracted from the audio outcome from each smart speaker microphone array, while the talker is in motion. To acquire these features, the system can undergo training via two primary methods: *real-world* data collection in actual rooms with people; or using *room simulations*. In the real-world data collection approach, training data would be collected by deploying smart speakers in real rooms where actual people are tracked and interact with the speakers. Conversely, in the room simulation method, the training data is collected by simulating a room layout configured with multiple smart speakers with microphone arrays, simulating various reverberation times to emulate authentic sound environments. The talker's movement is simulated by positioning simulated sound sources at different locations among the simulated speakers, from which audio features are extracted. Simulated environments offer the advantage of accumulating data that encompasses various smart speaker setups, room configurations, and various talker locations. This research leverages a training dataset generated from a simulated environment, and information about our simulation setup is available in V-A. This training data set was as close to real-world data as possible, using real speech; an additional, similar, data set was generated for testing the system.

B. SIP signalling for smart speaker session handover

The standard SIP protocol supports *personal mobility* to accommodate sessions on multiple devices, whether it occurs during *pre-call* or even *mid-call*. However, the current standards do not support seamless media transitions. In the context of *personal mobility*, it is possible to have several devices registered to a single SIP address for session management. Initiating a call session on multiple devices sharing the same SIP address can be accomplished through the existing SIP functionality, which supports either parallel or sequential call forwarding. This capability can be leveraged for dynamic session handover between the smart speakers. While a call is in progress, i.e during a *mid-call* scenario, and a more suitable smart speaker is identified for handover, based on the method proposed in IV-A, the session handover can be achieved using one of the methods below:

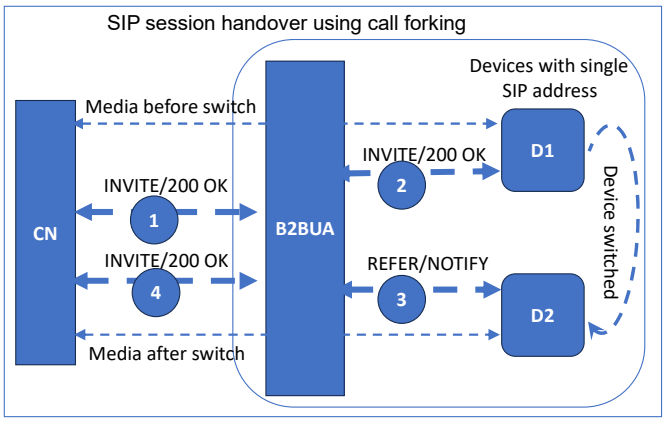


Fig. 4: SIP mid-call horizontal smart speaker session handover using REFER/NOTIFY between back-to-back user agent (B2BUA) and corresponding node (CN)

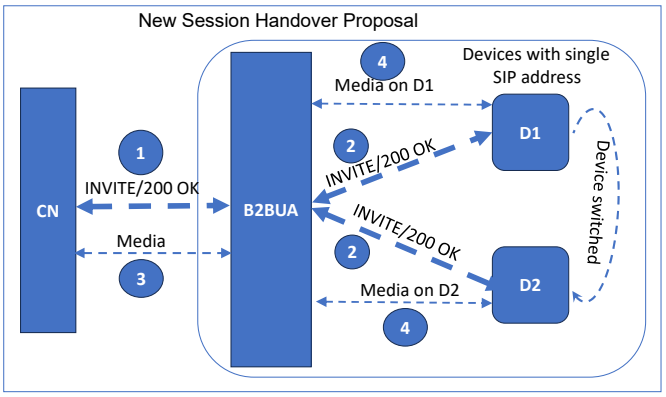


Fig. 5: SIP mid-call horizontal smart speaker session handover with modified back-to-back user agent (B2BUA) and corresponding node (CN)

Using existing SIP signalling: The existing session can be transferred to the new smart speaker using SIP Re-Invite or REFER methods. The message sequence flow at a higher level for this situation, involving *call sequential forking* and using REFER methods, is illustrated in Fig. 4. This method typically introduces a gap in the media while the signalling completes.

Using modified SIP signalling: This paper also introduces an alternative approach for session handover, as depicted in Fig. 5. Slight changes are required to the existing B2BUA behaviour. However, this novel method eliminates media interruptions and offers improved control over session management for smart speakers within the handover domain.

In practice media monitoring and switching is needed at a node in the network and it makes sense to carry this out on the B2BUA with the proposed modified SIP signalling which is straightforward to implement. In a practical deployment, this could be carried out at the home/office gateway or in one of the smart speakers, both of which are being used as a B2BUA in emerging systems.

TABLE I: Room simulation setup

Parameter	Description
Room Dimensions	7m X 5.5m X 2.4m
Room 1	D1: 1.0m X 3.5m X 0.9m D2: 5.5m X 3.5m X 0.9m
Room 2	D1: 1.5m X 3.5m X 0.9m D2: 6.0m X 3.5m X 0.9m
Source Positions	200 locus movements each with 50 time steps from 0.5m to 6m
RT60	0.8s, 1.5s
Audio features	All, C, C+A
Number of audio signals	50
Audio sample rate	16 kHz
Feature sample rate	0.2 s^{-1}

V. Results

The evaluation was designed to test the proposed method against a non-machine learning approach and to test the proposed custom-loss function against a commonly used ML loss function. The evaluation is performed across 20,000 different scenarios: 50 different speech signals (taken from [51]) were applied across 200 different sets of movement loci and two different room scenarios (Room 1 and Room 2). The speech samples and loci in the training set were different from those used in the testing set. Room 1 was used for the training and then testing was applied in both Room 1 and Room 2. This section describes the simulation environment used for the evaluation of the proposed ML-based technique as well as a comparative method that uses a simple predictor based only on the MSC as inspired by location-based techniques; although, there is no attempt to determine the actual locations. An additional comparison is made by using a standard mean squared error as a loss function for the ML technique. First, we explain the room simulation environment and the 1DCNN model configuration. Then, before presenting the main results, we show a set of graphs that illustrate some specific examples of the operation of the comparative methods and our proposed 1DCNN-based approach to understand the performance of the audio features.

A. Room simulation setup

Two rectangular rooms, detailed in Table I, were simulated using the Python package Pyroomacoustics [52]. These simulations emulate typical living spaces, though factors like varying wall absorbance, windows, and furniture were not included. The experiments are conducted in environments with realistic reverberation times (RT60) [53], replicating conditions found in real-world settings to mimic realistic sound environments. Within this typical room configuration, we simulate two smart speakers, (D1) and (D2), each with two omnidirectional microphones, positioned at opposite ends of the room with a 4.5-meter separation. This setup emulates an open-plan living area, with one smart speaker in the living room section and the other in the kitchen area.

The movement of the sound source, representing the talker's voice, is simulated using two methods, as shown in Fig. 6. The first is a simple linear movement, where the sound source moves in 0.05-metre steps across the room. The second is locus movement, where the source follows a pseudo-random path defined by three random positions connected by a cubic Bezier curve. For locus movement, 200 loci were generated, each sampled at 50 points along the curve at varying speeds to

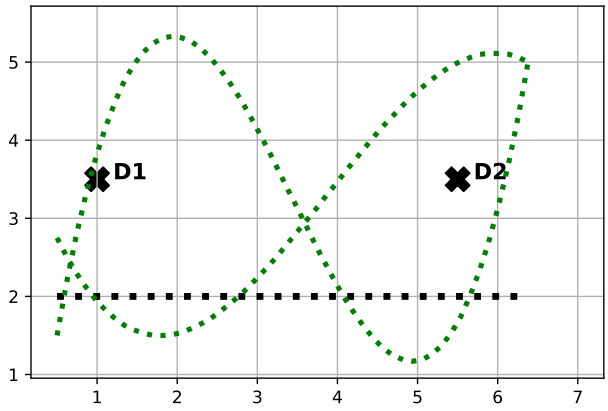


Fig. 6: Two examples of simulated movement: black is a linear move, green is one of the 200 simulated random motions using smoothed Bézier curves. The dimensions of the room are in meters and examples of smart speaker locations are shown

TABLE II: Hyperparameter tuned 1DCNN model

Parameter	Description
1st layer	filters: 120 kernel size: 45
2nd layer	filters: 124 kernel size: 30
3rd layer	filters: 112 kernel size: 15
dense layer	1
activation	ReLU
strides	1
padding	same
number of time steps	49
audio features	All, C, C+A

simulate different walking speeds. This represents the talker's movement between two locations, covering two smart speakers and manoeuvring around a third object, such as furniture.

B. Configuration of hyperparameter tuned 1DCNN model

The high-level 1DCNN model is depicted in Fig. 3 showing three convolutional layers that were optimised through hyperparameter tuning. The input tensor $\mathcal{T}$ is formed from column vectors consisting of the extracted features $\hat{\rho}_1$, $\hat{\rho}_2$, $A_{l,1}$, $A_{r,1}$, $A_{l,2}$, and $A_{r,2}$ ..., as described in Section IV-A2. The input tensor $\mathcal{T}$ then is fed through three consecutive 1DCNNs $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$. The parameters determined after hyperparameter are shown in Table II. A number of different audio feature sets were tested in different combinations including: MSC (abbreviated as just C); mean power spectral amplitude (A); MSC with amplitude (C+A); and, a combination of the former with the addition of MFCC, Mel spectral amplitude (All).

To produce a single predicted output, we conclude the model with a Dense layer consisting of one unit. This Dense layer takes inputs from the third layer in the model, computes a weighted sum of these inputs using a single set of weights, and generates a single predicted output.

C. Comparative techniques

As noted earlier we are interested to see how our approach compares against a simple MSC predictor and how the custom loss function improves the performance of the 1DCNN as described below.

1) Using simple MSC predictor: As a comparison technique, we use a double exponential smoothing predictor based on the MSC (Smoothed-MSC) as proposed by a number of location tracking applications and earlier work [5], [19], [24], [25]. The primary benefit of employing this method is that it does not need any calibration for smart speaker detection. Using this technique, the smart speaker exhibiting the higher MSC, denoted as $\max(\hat{\rho}_1, \hat{\rho}_2)$, is identified as the optimal choice for session transfer. The Monte-Carlo parameterization identified that values of $\alpha = 0.05$ and $\beta = 0.01$ are appropriate for the double exponential smoothing applied to the raw MSC values. The details of the outcome of this technique are described in Section V-D. We also tried Long Short-Term Memory (LSTM) as a predictor for the MSC values as a smoothing function but found it gave poorer results than the simple smoothing approach so we do not report that method here.

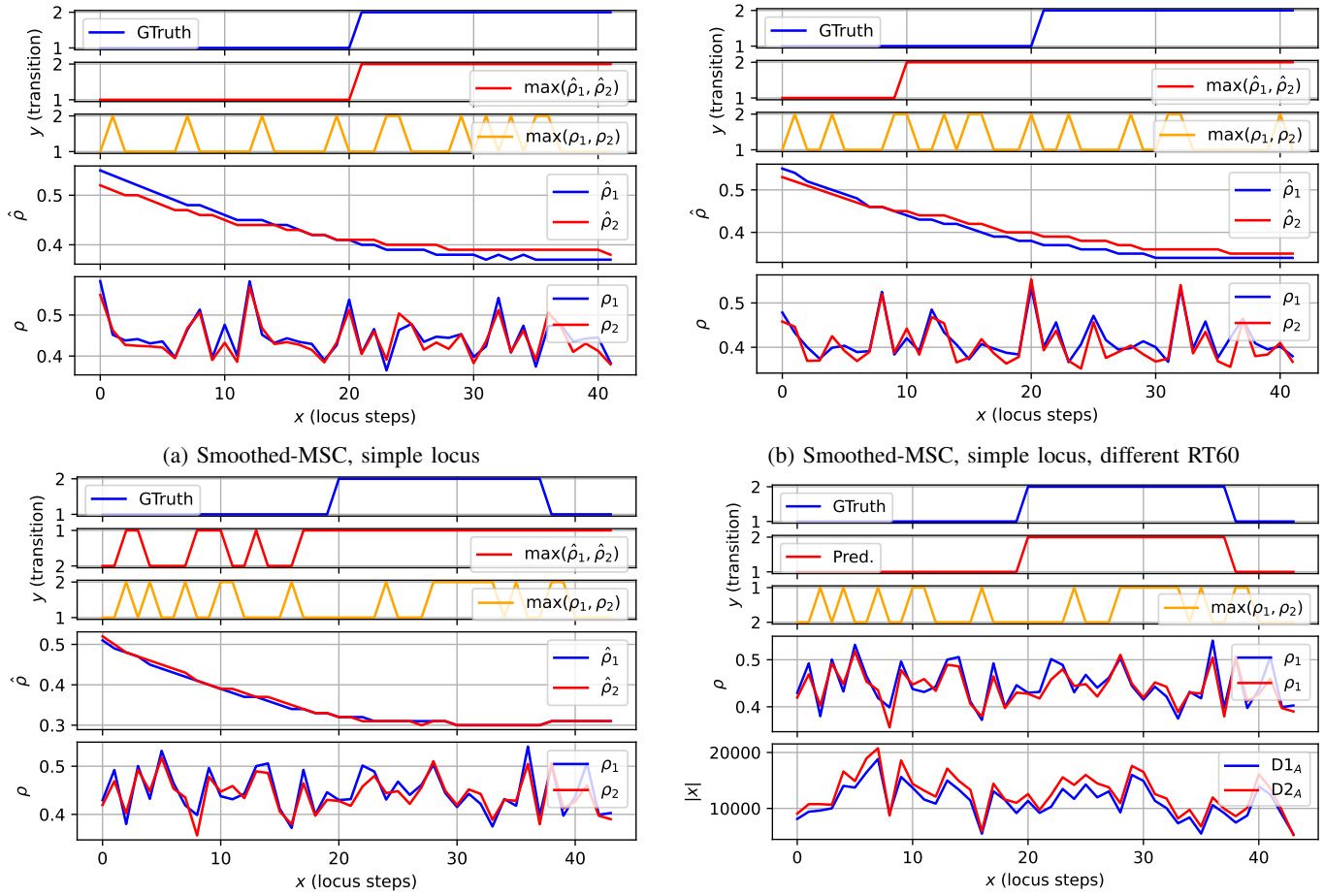
2) Using mean squared error as a loss function: As an additional comparison technique, we trained our 1DCNN model using a standard mean squared error (MSE) as a loss function. The outcomes of employing this technique will clearly highlight the issue outlined in Section III, emphasizing the need for a customized loss function as detailed in Section IV-A1.

D. Smart speaker prediction results

As noted earlier, we first here present some specific example graphs showing the operation of the Smoothed-MSC comparative method, shown in Fig. 7(a),(b),(c), and the 1DCNN approach shown in Fig. 7(d). The graphs show the ground truth (GTruth) as the top, blue, line as specified in (1) and the result of either the Smoothed-MSC ($\max(\hat{\rho}_1, \hat{\rho}_2)$) or the output of the 1DCNN (Pred.) as the second, red, line. Additionally, these graphs give more detail on the MSC used as a feature in the 1DCCN and as used in the Smoothed-MSC method. The MSC value is shown as the raw value from each smart speaker, ρ_1 and ρ_2 respectively, and, for the Smoothed-MSC comparative method, Fig. 7(a),(b),(c), the doubly exponentially smoothed values are also shown as $\hat{\rho}_1$ and $\hat{\rho}_2$ respectively.

The first graph, Fig. 7(a) depicts transitions based on the raw (non-smoothed) MSC values ($\max(\rho_1, \rho_2)$) and the Smoothed-MSC transitions ($\max(\hat{\rho}_1, \hat{\rho}_2)$) when there is a simple locus. This shows that transitions are not clearly detected using raw MSC, whereas transitions using the Smoothed-MSC approach closely approximate the ideal scenario. However, when this technique is applied to a different room scenario in Fig. 7(b) or more complex locus in Fig. 7(c) we see that the Smoothed-MSC fails to follow the desired performance as: it either transitions at the wrong locus position in Fig. 7(b); or, exhibits extraneous transitions in Fig. 7(c). This is likely to be caused by the predictor having fixed parameters (α and β) that do not encompass the wider set of scenarios such as real-world loci.

The graph in Fig. 7(d) illustrates the predicted transitions (Pred.) using the proposed 1DCNN with a custom loss function applied to the complex locus, closely approximating the ideal scenario during the transition. We only show one result for the proposed technique in this diagram to save



(a) Smoothed-MSC, simple locus

(b) Smoothed-MSC, simple locus, different RT60

(c) Smoothed-MSC, complex locus

(d) **Proposed 1DCNN-CL, complex locus**

Fig. 7: Sample result comparing the comparison technique, exponentially smoothed-MSC predictor in (a-c), with the proposed technique 1DCNN with custom loss (1DCNN-CL) (d). Note: 1DCNN-CL gave equally good results in the other, simple, cases, not shown here. The top (blue) line shows the ground truth, and the second (red) line shows the prediction of each technique. Appropriate features are shown below.

space as it worked also perfectly in the other cases. The graph displays raw MSC values (ρ_1, ρ_2) and signal magnitudes (D1$_A$, D2$_A$) captured at each locus position, which are the features employed in the 1DCNN predictor for forecasting the transition.

Furthermore, we performed a quantitative analysis of these techniques across diverse locus movements, various rooms, and different audio signals. The assessment of quantitative performance relies on two primary metrics:

number of extraneous transitions (ET): The count of extra (or missed) smart speaker transitions using a prediction compared to the number of smart speaker transitions using the ideal (ground truth) case. For example see Fig. 7(c) which has seven extraneous transitions (six before the ground truth and one failed after).

weighted normalised error (WNE): uses the loss function $L(t)$ to compute an error that takes into account the fact that: while an error near the ideal switching point is benign, an error further away from this ideal switching point becomes perceptually highly significant (see III).

Considering extraneous transitions as a metric is crucial since users generally would find unnecessary switches as very disruptive. Ideally, the number of extraneous transitions should be zero. The objective of WNE, is to calculate an error that increases when the selected smart speaker is substantially more distant than the ideal smart speaker and the ideal value should be zero.

The quantitative performance results shown in Table III highlight the poor performance of the comparative MSC technique on different room conditions and on random locus movement. However, Table IV shows that using our proposed 1DCNN solution significantly improves upon using the smoothed MSC approach, in particular exhibiting no extraneous transitions. Table IV also explores different feature choices and loss function by training the model with different feature combinations: solely MSC (C), absolute signal magnitude (A), MSC with A (C+A), and all audio features described in IV-A2 (All). We see that the custom loss (CL) function gives considerable improvement compared to a standard loss function such as MSE. We have chosen two results in Fig. 8 to demonstrate the effect of the custom loss function, as we see the 1DCNN-CL more closely tracks the ground truth based

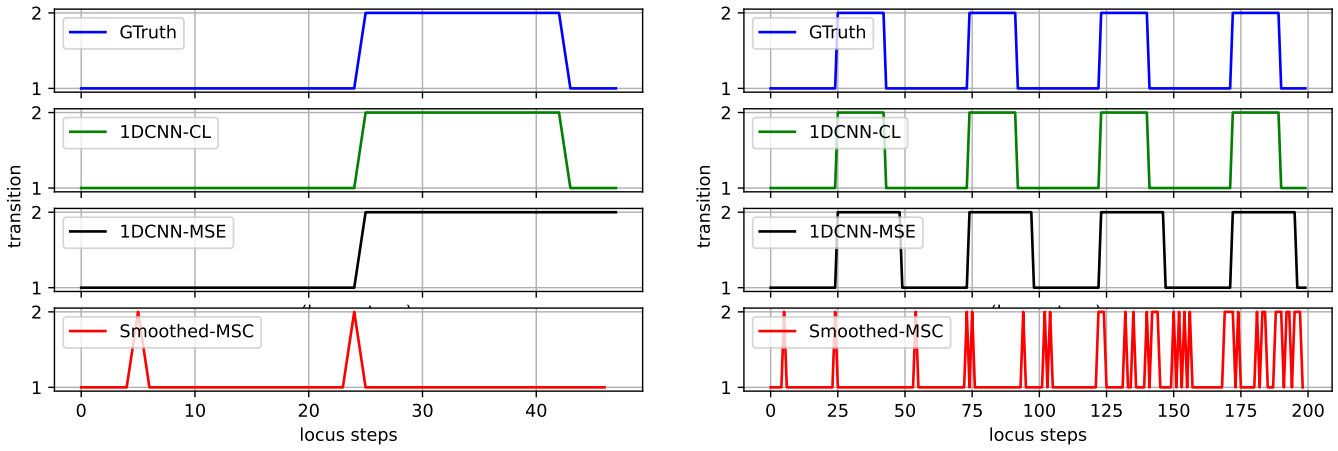


Fig. 8: Two samples from the results of using 1DCNN, showing comparative results for several locus positions, (a) and (b), using our proposed 1DCNN custom loss (1DCNN-CL) and the alternatives of either standard mean squared error as a loss function (1DCNN-MSE) or the exponentially smoothed MSC (Smoothed-MSC).

TABLE III: Performance of the double exponential smoothing (S) compared with the Non-smoothed (NS) approach with different locus types and room RT60 values showing it can work with a simple locus (straight line) but fails with more complex loci.

		Linear locus		Random locus	
Metric	NS/S	RT60-0.8	RT60-1.5	RT60-0.8	RT60-1.5
WNE	NS	0.43	0.59	0.43	0.45
	S	0	0.08	0.64	0.66
ET	NS	20	22	23	26
	S	0	0	7	9

upon the nearest smart speaker. Although the 1DCNN-CL does not transition exactly at the same point as the ground truth, as noted in IV-A, small differences are unlikely to be noticeable, however, with the standard MSE loss function (1DCNN-MSE) the smart speaker transition is much less accurate; indeed we see in Fig. 8 (a) that one of the transitions was not detected at all for the standard loss function (1DCNN-MSE).

To demonstrate that the trained model can work across multiple rooms, as well as loci it was not trained for, the model was trained on Room 1 and then tested on Room 2 with different smart speaker positions and RT60 times. Table V shows this comparison with highly different room types and RT, indicating that the model performs well in different and with highly reverberating rooms (RT60=1.5s).

The overall performance analysis concludes that:

- The simple MSC predictor (Smoothed-MSC) offers a straightforward solution for detecting the appropriate smart speaker for handover, but it does not address the complexities of real-world loci scenarios.
- Using MSE as the loss function to train the 1DCNN model for smart speaker detection, improves the performance over the simple MSC predictor for loci of increased complexity, but is less accurate than the proposed solution and leads to missing/extraneous transitions
- The proposed **1DCNN-CL** model, trained with both the absolute signal magnitude and the MSC, addresses all complex real-world loci scenarios and produces optimal results – without including additional features such as the Mel magnitude spectrum or MFCCs.

TABLE IV: Performance of proposed 1DCNN with custom loss (1DCNN-CL) with various features and compared against exponential smoothing (Smoothed-MSC) and a 1DCNN with a standard loss (1DCNN-MSE). Features: Mean spectral magnitude (A), MSC (C), multiple features (All). Complex loci were used for all 20,000 results.

Model	Features	WNE	ET
Smoothed-MSC	C	0.86	4783
1DCNN-MSE	C+A	0.22	1
1DCNN-CL	All	0.087	0
1DCNN-CL	C	0.05	0
1DCNN-CL	**C+A**	**0.03**	**0**

TABLE V: Performance of the proposed 1DCNN with custom loss (1DCNN-CL) using C+A features and with varied room types, Room 1 (RT1) and Room 2 (RT2) with various RT60 values.

Train RT1	Test RT2	WNE	ET
0.8s	0.8s	0.033	0
0.8s	1.5s	0.039	0
1.5s	0.8s	0.044	0

VI. Discussion

The solution presented in this paper is, to the knowledge of the authors, the first smart speaker handover mechanism for smart speech handover between smart speakers that does not require precise smart speaker calibration using location. This is a significant advance but clearly, the proposal of the paper would require some future engineering work to create a fully working system. We consider some of these aspects here first considering performance and then other practical issues.

A. Performance evaluation

The system described needs to operate in consumer smart speakers which have modest computational power and sometimes operate purely on battery power. The solution has been designed to only require modest computational power. Each convolution layer has $\mathcal{O}(nkfd)$ [8] complexity where n is the input size, k is the kernel size, f is the number of filters, and, d is the number of channels (number of features). Using the hyperparameter tuned model shown in Table II we thus find that for each time sample the complete 1DCNN model would require of the order 0.7 M operations for each locus point. As the averaged features are sampled 5 times per second, this gives the total number of floating point operations as 3.5 M/s which is well within the capability of a small modern embedded processor such as would be found in a consumer smart speaker.

B. Practical considerations

This section considers future engineering work required to implement a full consumer application. We have shown that the proposed method works with some different room sizes and reverberation characteristics in a simulated environment that closely approximates real-world scenarios. For a robust solution, the machine learning would need to be optimised across a much wider set of examples. For example, while the simulation approach used in this paper shows good proof of principle, further training in real environments would help train the system across a wider set of conditions such as different rooms and different smart speakers. Training the system in real rooms would require a method of tracking a talker while taking part in the training sessions, although no such tracking would be required in the operational phase. Advanced techniques for user tracking using video have been proven in other fields for machine learning purposes. For example, the use of eye-tracking to record user interactions with Video-on-Demand (VoD) applications to understand user experience [54]. Advanced video-based user tracking techniques could be employed to provide the required ground-truth inputs from training sessions with minimal operational effort.

Although the simulation and training in the paper were very close to real-world conditions, the training in this paper was specific to the microphone arrangement for the specific smart speakers used for the results. For an engineering solution it is likely that retraining would be required for a new smart speaker with different microphone configurations for example with different distances between the microphones. As this retraining would be carried out by a vendor, before sale, it would still allow consumers to use the system without recalibration.

C. Future work

In the work of this paper, only one talker was used in the simulations. While this is a good scenario for the consumer use case of this paper, as often there is one dominant talker in a location, this would be less valid in situations where there are multiple talkers in a single auditory environment such as business meetings with maybe only one or two remote members. It may be in such cases the switching of the talker input proposed in this paper would be useful to use the smart speaker closest to the talker, but switching the played back speech into the room may not benefit from switching as it would make the remote talker's apparent location in the room jump from one smart speaker to another. Further work on this user experience for different meeting types is thus required, a possible solution combines the work of this paper with blind-source separation techniques [55].

Finally, we commented in III that we assumed the smart speaker nearest the user would be the ideal smart speaker. The use of coherence in the approach is an interesting feature. We conjecture that in a room that consists of a portion with a dead acoustic environment (such as that set up for living space with carpets) and a portion with a live acoustic environment (such as a kitchen with hard surfaces), it would be beneficial to switch to a smart speaker in the dead environment earlier than that in the live environment. Thus, it might be that the use of a coherence-based ground truth rather than distance-based ground truth will result in improved perceptual performance but this would require further perceptual testing that is out of the scope of this paper.

VII. Conclusion

This paper addresses the need to autonomously handover speech between smart speakers during a voice call. This is important for the next generation of smart speakers so that they can inter-operate with emerging applications that enable users to interact with applications seamlessly and without manual intervention. The work identifies the appropriate smart speaker from multivariate audio features using a 1DCNN and the utilisation of the SIP protocol for session transition. The research addresses challenges associated with employing 1DCNN as a classifier for smart speaker prediction and suggests a regression-based solution incorporating a custom loss function. The results indicate that using 1DCNN as a regressor with a custom loss function eliminates unnecessary transitions and improves transition accuracy compared to a system that only uses the audio features themselves. This research also proposes modified SIP signalling to enable uninterrupted media transition. To validate the proposed 1DCNN with a custom loss function, extensive testing was conducted in an environment that simulated realistic conditions featuring diverse room conditions, varied talker movements, and a range of speech signals. This demonstrated that automatic switching between smart speakers in a home is possible without requiring exact calibration of the location and orientation of the smart speakers. This is important for achieving, future, intelligent home devices that can act autonomously for consumers [2].

Acknowledgment

The authors would like to acknowledge Ian Kegel from BT who contributed to the early concepts of this work but, sadly, his untimely loss made it impossible for him to contribute to the manuscript as an author.

REFERENCES

[1] M. Philpott. (2021, January) The Future Telco Connected Home. [Online]. Available: https://www.broadband-forum.org/

[2] C. K. Wu, C.-T. Cheng, Y. Uwate, G. Chen, S. Mumtaz, and K. F. Tsang, "State-of-the-Art and Research Opportunities for Next-Generation Consumer Electronics," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 4, pp. 937–948, 2023.

[3] "The Foundation for Connected Things". [Online]. Available: https://csa-iot.org/all-solutions/matter/

[4] "https://www.globalservices.bt.com/en/insights/whitepapers/what-makes-a-smart-home-smart". [Online]. Available: https://www.globalservices.bt.com/en/insights/whitepapers/what-makes-a-smart-home-smart

[5] S. Vesa, "Binaural Sound Source Distance Learning in Rooms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 8, pp. 1498–1507, 2009.

[6] V. N. Mitnala, M. J. Reed, I. Kegel, and J. Bicknell, "Avoiding handover interruptions in pervasive communication applications through machine learning," in *2021 IEEE International Conference and Expo on Real Time Communications at IIT (RTC)*, 2021, pp. 1–8.

[7] B. Kapralos, M. Jenkin, and E. Milios, "Audiovisual localization of multiple speakers in a video teleconferencing setting," *International Journal of Imaging Systems and Technology*, vol. 13, pp. 95 – 105, 07 2002.

[8] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," *Mechanical Systems and Signal Processing*, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0888327020307846

[9] H. Kim, S. Hwang, J. Kim, and Z. Lee, "Toward Smart Communication Components: Recent Advances in Human and AI Speaker Interaction," *Electronics*, vol. 11, no. 10, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/10/1533

[10] A. Ahrens, K. D. Lund, M. Marschall, and T. Dau, "Sound source localization with varying amount of visual information in virtual reality," *PLoS ONE*, vol. 14, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:88480033

[11] K. Ma, H. Chen, Z. Wu, X. Hao, G. Yan, W. Li, L. Shao, G. Meng, and W. Zhang, "A wave-confining metasphere beamforming acoustic sensor for superior human-machine voice interaction," *Science Advances*, vol. 8, no. 39, p. eadc9230, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/sciadv.adc9230

[12] C. Rascon and I. Meza, "Localization of sound sources in robotics: A review," *Robotics and Autonomous Systems*, vol. 96, pp. 184–210, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889016304742

[13] M. M. Faraji, S. B. Shouraki, E. Iranmehr, and B. Linares-Barranco, "Sound Source Localization in Wide-Range Outdoor Environment Using Distributed Sensor Network," *IEEE Sensors Journal*, vol. 20, no. 4, pp. 2234–2246, 2020.

[14] C. Evers, H. W. Löllmann, H. Mellmann, A. Schmidt, H. Barfuss, P. A. Naylor, and W. Kellermann, "The LOCATA Challenge: Acoustic Source Localization and Tracking," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1620–1643, 2020.

[15] S. E. Chazan, H. Hammer, G. Hazan, J. Goldberger, and S. Gannot, "Multi-Microphone Speaker Separation based on Deep DOA Estimation," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[16] A. Xenaki, J. Boldt, and M. Christensen, "Sound source localization and speech enhancement with sparse Bayesian learning beamforming," *The Journal of the Acoustical Society of America*, vol. 143, pp. 3912–3921, 06 2018.

[17] H. Lim, I.-C. Yoo, Y. Cho, and D. Yook, "Speaker localization in noisy environments using steered response voice power," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 1, pp. 112–118, 2015.

[18] T. Kim, H. Park, S. H. Hong, and Y. Chung, "Integrated system of face recognition and sound localization for a smart door phone," *IEEE Transactions on Consumer Electronics*, vol. 59, no. 3, pp. 598–603, 2013.

[19] K. Zhagyparova, R. Zhagypar, A. Zollanvari, and M. T. Akhtar, "Supervised Learning-based Sound Source Distance Estimation Using Multivariate Features," *TENSYMP 2021 - 2021 IEEE Reg. 10 Symp.*, pp. 1–5, 2021.

[20] A. Brendel and W. Kellermann, "Learning-based acoustic source-microphone distance estimation using the coherent-to-diffuse power ratio," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2018-April, pp. 61–65, 2018.

[21] Y. Li and H. Chen, "Reverberation Robust Feature Extraction for Sound Source Localization Using a Small-Sized Microphone Array," *IEEE Sensors Journal*, vol. 17, no. 19, pp. 6331–6339, 2017.

[22] Z. K. Abdul and A. K. Al-Talabani, "Mel Frequency Cepstral Coefficient and its Applications: A Review," *IEEE Access*, vol. 10, pp. 122 136–122 158, 2022.

[23] K. Baimirov, E. Mergengali, and B. Baimirov, "Overview of the latest research related to smart speakers," in *2022 IEEE 7th International Energy Conference (ENERGYCON)*, 2022, pp. 1–5.

[24] P. Calamia, N. Balsam, and P. Robinson, "Blind estimation of the direct-to-reverberant ratio using a beta distribution fit to binaural coherence," *The Journal of the Acoustical Society of America*, vol. 148, no. 4, pp. EL359–EL364, 10 2020. [Online]. Available: https://doi.org/10.1121/10.0002144

[25] V. N. Mitnala, M. J. Reed, I. Kegel, and J. Bicknell, "Seamless device handover for pervasive speech communication," in *2022 5th International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, 2022, pp. 1–6.

[26] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, 2015.

[27] S. Kobayashi, T. Hasegawa, T. Miyoshi, and M. Koshino, "MarNASNets: Toward CNN Model Architectures Specific to Sensor-Based Human Activity Recognition," *IEEE Sensors Journal*, vol. 23, no. 16, pp. 18 708–18 717, 2023.

[28] H. Cho and S. M. Yoon, "Divide and Conquer-Based 1D CNN Human Activity Recognition Using Test Data Sharpening," *Sensors*, vol. 18, no. 4, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/4/1055

[29] A. Chowdhury and A. Ross, "Fusing MFCC and LPC Features Using 1D Triplet CNN for Speaker Recognition in Severely Degraded Audio Signals," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1616–1629, 2020.

[30] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[31] X. Wang, M. Xia, and W. Deng, "MSRN-Informer: Time Series Prediction Model Based on Multi-Scale Residual Network," *IEEE Access*, vol. 11, pp. 65 059–65 065, 2023.

[32] A. Lawal, S. Rehman, L. M. Alhems, and M. M. Alam, "Wind Speed Prediction Using Hybrid 1D CNN and BLSTM Network," *IEEE Access*, vol. 9, pp. 156 672–156 679, 2021.

[33] F. Shen, J. Liu, and K. Wu, "Multivariate Time Series Forecasting Based on Elastic Net and High-Order Fuzzy Cognitive Maps: A Case Study on Human Action Prediction Through EEG Signals," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 8, pp. 2336–2348, 2021.

[34] M. Aazam, S. u. Islam, S. T. Lone, and A. Abbas, "Cloud of Things (CoT): Cloud-Fog-IoT Task Offloading for Sustainable Internet of Things," *IEEE Transactions on Sustainable Computing*, vol. 7, no. 1, pp. 87–98, 2022.

[35] Y. Ma, Y. Zhang, Z. Sheng, H. Ruan, J. Wang, and Y. Sun, "CGMP: cloud-assisted green multimedia processing," *Multimedia Tools and Applications*, vol. 75, pp. 13 317–13 332, 2016. [Online]. Available: https://doi.org/10.1007/s11042-015-2783-2

[36] A. B. Johnston, *SIP: understanding the session initiation protocol*. Artech House, 2015.

[37] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "RFC3261: SIP: session initiation protocol," 2002.

[38] M. A. Qadeer, A. H. Khan, J. A. Ansari, and S. Waheed, "IMS network architecture," *Proc. - 2009 Int. Conf. Futur. Comput. Commun. ICFCC 2009*, pp. 329–333, 2009.

[39] G. Carella, M. Corici, P. Crosta, P. Comi, T. M. Bohnert, A. A. Corici, D. Vingarzan, and T. Magedanz, "Cloudified IP Multimedia Subsystem (IMS) for Network Function Virtualization (NFV)-based architectures," in *2014 IEEE Symposium on Computers and Communications (ISCC)*, vol. Workshops, 2014, pp. 1–6.

[40] J. Whiteaker, F. Schneider, R. Teixeira, C. Diot, A. Soule, F. Picconi, and M. May, "Expanding home services with advanced gateways," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 5, p. 37–43, Sep. 2012. [Online]. Available: https://doi.org/10.1145/2378956.2378962

[41] V. K. Gurbani and K. Q. Liu, "Session initiation protocol: Service residency and resiliency," *Bell Labs Technical Journal*, vol. 8, no. 1, pp. 83–94, 2003.

[42] S. V. Azhari, M. Homayouni, H. Nemati, J. Enayatizadeh, and A. Akbari, "Overload control in SIP networks using no explicit feedback: A window based approach," *Computer Communications*, vol. 35, no. 12, pp. 1472–1483, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366412001260

[43] D. Y. Yavas, I. Hokelek, and B. Gunsel, "Modeling of Priority-based Request Scheduling Mechanism for Finite Buffer SIP Servers," in *Proceedings of the 11th International Conference on Queueing Theory and Network Applications*, ser. QTNA '16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/3016032.3016055

[44] R. Gandotra and L. Perigo, "SDVoIP—A Software-Defined VoIP Framework For SIP And Dynamic QoS," *The Computer Journal*, vol. 64, no. 1, pp. 254–263, 2019.

[45] H. Schulzrinne and E. Wedlund, "Application-layer mobility using SIP," *ACM SIGMOBILE mobile computing and communications review*, vol. 4, no. 3, pp. 47–57, 2000.

[46] H. Schulzrinne, "Personal mobility for multimedia services in the Internet," in *European Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services*. Springer, 1996, pp. 143–161.

[47] Y.-j. Oh, E.-h. Paik, and K.-r. Park, "Design of a SIP-based real-time visitor communication and door control architecture using a home gateway," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 4, pp. 1256–1260, 2006.

[48] D. Pinardi, A. Toscani, M. Binelli, L. Saccenti, A. Farina, and L. Cattani, "Full-Digital Microphone Meta-Arrays for Consumer Electronics," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 3, pp. 640–648, 2023.

[49] M. Slaney, "Auditory Toolbox: A MATLAB Toolbox for Auditory Modeling Work. Technical Report, version 2, Interval Research Corporation," Interval Research Corporation, Tech. Rep., 1998. [Online]. Available: https://engineering.purdue.edu/~malcolm/interval/1998-010/

[50] L. Datta, "A Survey on Activation Functions and their relation with Xavier and He Normal Initialization," 2020.

[51] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[52] R. Scheibler, E. Bezzam, and I. Dokmanic, "Pyroomacoustics: A Python Package for Audio Room Simulation and Array Processing Algorithms," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, apr 2018. [Online]. Available: https://doi.org/10.1109%2Ficassp.2018.8461310

[53] H. Kuttruff, *Room acoustics*. CRC Press, 2016.

[54] Z. Chen, S. Zhang, S. McClean, F. Hart, M. Milliken, B. Allan, and I. Kegel, "Process Mining IPTV Customer Eye Gaze Movement Using Discrete-Time Markov Chains," *Algorithms*, vol. 16, no. 2, 2023. [Online]. Available: https://www.mdpi.com/1999-4893/16/2/82

[55] S. Ansari, K. A. Alnajjar, T. Khater, S. Mahmoud, and A. Hussain, "A Robust Hybrid Neural Network Architecture for Blind Source Separation of Speech Signals Exploiting Deep Learning," *IEEE Access*, vol. 11, pp. 100 414–100 437, 2023.