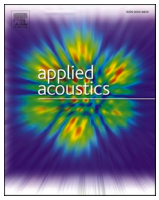
Applied Acoustics 240 (2025) 110954

Contents lists available at ScienceDirect

Applied Acoustics

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



Single channel speech denoising by DDPG reinforcement learning agent

Sania Gul [a,1], Muhammad Salman Khan [b,1,*]

[a] Department of Electrical Engineering, University of Engineering and Technology, Peshawar, Pakistan
[b] Department of Electrical Engineering, College of Engineering, Qatar University, Doha, Qatar

ARTICLE INFO

Keywords:
Speech denoising
Reinforcement learning
DDPG agent
TD3 agent
Reward function
Time–frequency masking

ABSTRACT

Speech denoising (SD) covers the algorithms that suppress the background noise from the contaminated speech and improve its clarity. In this paper, a novel SD algorithm is presented based on the deep deterministic policy gradient (DDPG) agent; an off-policy reinforcement learning (RL) agent with a continuous action space. The noisy speech is first converted from the time domain to the time–frequency (TF) domain by taking its short-time Fourier transform (STFT), and then two separate DDPG agents are trained on the magnitude and phase components of the STFT. The reward function used for training these agents is the relative perceptual quality score of speech. After training, the DDPG agents generate the magnitude and phase soft masks, when noisy speech is given as input to them. These masks are then applied to the complex STFT matrix of the noisy speech to obtain the denoised speech. For matched testing data, the proposed system offers an improvement of 1.55 points in the perceptual evaluation of speech quality (PESQ) over the unprocessed speech, the highest among the other recent state-of-the-art models used for comparison in this paper. It achieves this performance by utilizing data that is 7 times smaller than that required by other models. Also, its learnable parameters are the lowest among all models, almost 12 times less than the next most compact model, based on another continuous RL agent (policy gradient (PG)) for estimating its convolutional kernels. When cascaded with a coloured spectrogram-based SD model, the proposed model further improves the PESQ by 0.07, CSIG by 1.23, and COVL by 1.4 points; the metrics estimating respectively the perceived quality of speech, its composite distortion, and its composite overall quality, surpassing all other baseline systems compared here. In the cascaded configuration, our proposed model offers the highest gain in PESQ (by 0.78 points) at 50 times fewer episodes, compared to the already trained speech enhancement and recognition models utilizing discrete agents for their performance improvement.

1. Introduction

Speech denoising (SD) comes under the umbrella of speech enhancement (SE) algorithms and is used for removing the unwanted background noise that degrades speech quality. It is not only required by people with hearing problems, but it is also an integral part of multimedia intelligent systems requiring man–machine interaction, where the noise can severely degrade the computer-based speech recognition accuracy [1]. For people with normal hearing, SD is required to understand the speech contents recorded in noisy backgrounds e.g. roads, seashores, bus and railway stations, and working factories [2].

Since its inception in the mid-20th century, deep neural networks (DNNs) have generated astonishing results in the SE applications of source separation, denoising, and dereverberation [3]. Broad categorization of the deep learning domain results in three classes: a) supervised learning (SL), b) unsupervised learning (USL), and c) reinforcement learning (RL) [4]. The SL models of audio enhancement can retrieve the speech contaminated by noise either by 1) predicting a mask that filters out noise when applied to the noisy speech, or 2) by mapping the noisy speech directly to a cleaner one [5]. U-Net, generative adversarial networks (GANs), and recurrent neural networks (RNNs) are supervised networks commonly used for SE. The supervised learning offers incredible performance gains compared to the previous signal processing and machine learning methods but they fail to generalize well in the unseen acoustic conditions (e.g. different reverberation characteristics of the recording area, different recording equipment, source-to-microphone spacing and orientation), noise types and signal-to-noise (SNR) levels. Also, they have a stringent requirement of carefully aligned parallel noisy-clean datasets. In contrast, the USL has relaxed the degree of supervision required for model training and improved the

* Corresponding author.
E-mail address: salman@qu.edu.qa (M.S. Khan).
[1] Equal contribution.

https://doi.org/10.1016/j.apacoust.2025.110954
Received 21 November 2024; Received in revised form 31 May 2025; Accepted 14 July 2025
Available online 19 July 2025

generalization capability of SE models to various possible acoustic conditions and noise types. Likewise, they have also eased the design of the training/ testing dataset [6] by directly using in-the-wild noisy data without any access to the clean data [7]. Autoencoders and their variants and diffusion-based generative models [8] are examples of unsupervised networks used for SE [6]. However, the performance of USL is not on par with the SL algorithms [9].

Keeping in view the pros and cons of SL and USL, a new set of models based on self-supervised learning (SSL) is proposed for SE [10], which merges the USL with SL to obtain the benefits of both types. In SSL, the network learns suitable representations from a large-scale unlabelled dataset (e.g., noisy speech) in an unsupervised manner (called the upstream or pretext task [11]) and later applies them through generalization to the downstream tasks. The pseudo-labels for the upstream task are automatically generated by the network based on some attributes of training data [12]. This arrangement allows the model to learn autonomously in various acoustic conditions without human intervention, thereby overcoming the drawbacks and constraints of SL [10]. Moreover, much less data is required to fine-tune the SSL model for the downstream tasks than is required by the pure supervised models to achieve similar or better performance [11].

The third class of deep learning i.e. RL is inspired by animal learning. It does not require prior knowledge and obtains its optimal policy by trial-and-error (experience) and its continuous interaction with the dynamic environment. RL has made possible crafting artificial intelligent systems that are responsive and can effectively learn from their environment [13]. The potential of online learning and self-improving makes RL one of the most favourite intelligent agent's core technologies [14]. The use of deep learning in RL has accelerated its progress, introducing a new class in RL, called deep reinforcement learning (DRL). The use of DNNs for RL enables it to scale to high-dimensional state and action spaces for decision-making problems that were previously intractable [13]. The core of DRL is experience-driven learning. An agent interacts with its environment, and upon observing the consequences of its actions in the form of reward or penalty, changes its behaviour. The reward signal provided by the environment in RL is a kind of appraisal of the action by the agent, but it does not tell the agent how to generate the correct action [14]. The best sequence of action (known as policy) is determined by the rewards that are received by the agent as feedback from the environment [13]. The goal of the agent is to learn a policy that maximizes the overall cumulative reward [15].

The agent can learn either by directly interacting with its environment (online policy agent) or from an offline database (also called an experience buffer) of previously collected experiences. Such agents are called offline policy/ batch/ data-driven agents. Both types of RL agents continuously interact with the environment to update their policy. Learning the policy offline is typically preferred due to the following reasons: 1) the availability of large training times in offline learning to learn the best behaviour [15], 2) for improved generalization in complex domains [16], 3) to avoid the nonsensical behaviour due to explorations during training [15], and 4) when the online interactions are impractical and costly [17]. After learning the policy in an offline manner, the agent can be tuned online with the added advantage that the initial actions are safer and cheaper than the randomly adopted policy [16]. Q-Learning, deep Q-network (DQN), deep deterministic policy gradient (DDPG), twin-delayed deep deterministic (TD3) policy gradient, soft actor-critic (SAC) are offline agents, whereas SARSA, REINFORCE policy gradient (PG), actor-critic (AC), and proximal policy optimization (PPO) are few examples of online policy agents [18].

DRL is mainly used in control systems, robotics, indoor navigation, video gaming, computer vision, natural language processing, transportation, and dispatch management [14]. In contrast to SL and USL, the use of RL for audio applications e.g. automatic speech recognition (ASR), music generation, speech emotion recognition, spoken dialogue systems, and audio-driven robotics is recent and took off in 2015 [15]. Till now it has been used on a very limited scale for audio enhancement, particularly speech enhancement (SE) [15].

Supervised learning needs a labelled dataset, which sometimes is costly or even impossible to obtain. RL allows the autonomous system to learn from their experiences rather than exclusively under the supervision of a 'knowledgeable' teacher [19]. Using algorithms e.g. model-agnostic *meta*-learning framework (MAML) [20]) with reinforcement learning in [21] has proved to reduce the data required for training to only a few shots.

In the case of SE by DNNs, there is a mismatch between the final evaluation metrics used to judge the performance of a system and the objective function used during training to adjust the weights of the network [22]. The objective function mostly used for DNN optimization is mean square error, the value of which may not be indicative of human perception. Directly using the quality metrics as an objective function may be inappropriate for DNNs due to: 1) the metrics being non-differentiable (e.g. perceptual evaluation of speech quality PESQ ([23;24])), and 2) extremely complex correlation existing between the input and output of DNNs. Simplification of metrics to make them differentiable requires high expertise in the relevant application. On the other hand, the use of neural metrics (e.g. Quality-Net or GANs) as an objective function during the DNN training is not feasible as in the former, after a few iterations, the estimated quality score usually goes in opposition to the true quality, while training the latter is itself a complex task [25]. RL is the solution to implement the non-differentiable objective functions (e.g. PESQ) that correlate well with the human perception of speech quality [25]. In RL terms, the reward produced by the environment, after an action taken by the agent, is called a step-reward [26]. One of the main motivations to use RL for SE models is that it can use the objective metrics as its step-reward to judge its progress towards the goal of achieving better speech quality. Although it is instinctive to use the perceptual score (e.g. PESQ) directly as a reward for each step, it is not preferred due to being affected by the performance of the RL-based SD model, and also by the SNR level of the input signal [22]. So to avoid the external conditions affecting the step-reward, the SE by RL schemes proposed in [22] and [25] estimate the reward function by taking the difference between two perceptual scores. This relative reward function is inspired by the victory/ defeat in the gaming model [27] and generates a perceptually better-quality speech than the supervised models as proved in [22] and [25].

1.1. Related Work

The first SD model to use RL is [22], where it is used to self-optimize an existing SE system, which itself is a deep feed-forward neural network, pretrained on time–frequency (TF) masks of clean and noisy data, using the discriminative pretraining mechanism [28], and later self-optimized by a Q-learning agent (with discrete action space) using the relative PESQ score as its step-reward. The action space for the RL agent is estimated from the K-Means of the ideal masks. The step-reward is calculated by subtracting the PESQ score of the speech estimated by the RL agent from the PESQ score of the speech estimated by another deep neural network, already trained over the ideal masks estimated by Wiener filter [22], which however requires the availability of separate speech and the noise signals, unavailable in our selected dataset [29] used for experiments.

The noise suppression algorithm [30] uses a Deep Q-Network (DQN) RL agent (having discrete action space) designed over an LSTM architecture to tune the control parameters of an existing SE algorithm [30] to enable it to robustly adapt itself to the variations in the incoming noise. For step-reward calculation, the model uses the negated mean square error difference between the clean and estimated speech frames, directly in the time domain.

The SE model [31] designed specifically for the hearing aid device uses a DQN agent (designed using the CNN architecture), which uses the user's feedback to select one of the four discrete actions, depicting the user preferences of compression ratios to be applied on different

frequency bands of the incoming speech and music. The step-reward is the loss function of probabilities of the user's preference of any compression ratio over the others.

The SE model [25] directly processes the time domain signal and uses an RL agent to predict the mean and variance of the multivariate Gaussian distribution, from which the convolution kernel for the dynamic filter is sampled. The proposed model [25] uses the policy gradient (PG) agent (with continuous action space) to achieve its objective. In [25], the PESQ score of noisy speech is subtracted from the PESQ score of the estimated speech and if the result is negative the step-reward is set to zero to avoid negative values.

The automatic speech recognition (ASR) model [32] utilizes an RL agent in its front-end SD module to improve the back-end ASR performance. The SD module of this ASR model uses the same RL-based algorithm as proposed in [22].

The SE model [21] also uses the model proposed in [22] for initialization of the model-agnostic *meta*-learning framework (MAML) [20], an algorithm allowing deep neural networks to adapt quickly to a new task, using its prior *meta*-knowledge, requiring only a few samples of that task for adaptation.

Given the continuous nature of the soft masks required for SD, in this paper a novel SD algorithm using a DDPG agent having a continuous action space is proposed. As in [22], the reward function for the proposed model is also based on a perceptually relevant quality metric PESQ. To the best of our knowledge, this is the first time that the DDPG agent has been used for the SD task. It has been used previously for active noise cancellation (ANC) to cancel the narrowband noise [33], produced by household appliances, automotive and manufacturing industries, using mechanical and industrial equipment such as blowers, engines, fans, transformers, and compressors [34].

1.2. Our Contribution

The main contributions of our proposed model are summarised below:

1. It is the first attempt to achieve SD by a DDPG agent, which significantly reduces the computational cost compared to the discrete agents used in models [22,30,32], and [21]
2. Unlike the models [22,30,32], and [21], where an RL agent is just utilized to further improve the performance of already existing models, our proposed model is a completely autonomous SE model in its own right. However, it can also be cascaded with other state-of-the-art models to improve their performance
3. Unlike the masking-based models [22,30,32], and [21], where the network is bound to choose the mask from an already derived set of masks, our proposed model learns the appropriate mask only from its experiences

The rest of the paper is organized as follows. In the next section, an overview of our proposed SE algorithm is given. In Section 3, the dataset used, network settings, evaluation metrics, and the baseline algorithms used for comparison are described. Experimental results are presented in Section 4 and the limitations are discussed in Section 5. The paper is concluded in Section 6.

2. System Overview

In case of noisy speech, the time domain signal $x(t)$ is composed of two components: 1) clean speech $s(t)$, and 2) ambient noise $n(t)$. It is given mathematically as:

$$x(t) = s(t) + n(t) \tag{1}$$

Our proposed model is named 'SED', 'Speech Enhancement by DDPG agent'. It is a masking-based SD algorithm. Two separate SED models are trained. They are respectively called SED_{mag} and SED_{pha}. The first one estimates the magnitude mask and the second is designed to estimate the phase mask. The training and testing schemes used for SED are described below.

2.1. Testing Scheme

Fig. 1 shows the block diagram of our proposed SD algorithm during its testing phase.

Similar to the supervised TF masking-based SD systems e.g. [35,36,37], and [38], our proposed model SED also takes the noisy speech signal $x(t)$, convert it to TF domain by taking its short-time-Fourier-Transform (STFT) and separates the magnitude and phase components in two different matrices *Mag* and *Pha* respectively. The time frames (the columns) in these matrices are given in sequence as an input to SED_{mag} and SED_{pha}, which estimate the magnitude and phase soft masks respectively. These masks are again arranged as matrices M_{mag} and M_{pha}. These matrices are then applied to the complex STFT matrix of the noisy speech to obtain the spectrogram of the denoised speech $\widehat{S}$ as shown in Fig. 1 and given in eq. (2) as:

$$\widehat{S}(\omega,k) = X(\omega,k).\times M_{mag}(\omega,k).\times M_{pha}(\omega,k) \tag{2}$$

where $\omega = \{1,2,\cdots\cdots\Omega\}$ is the frequency index and $k = \{1,2,\cdots\cdots,K\}$ is the time index, '$.\times$' is the Hadamard (element-wise) multiplier, $X(\omega, k)$ is the complex STFT of the noisy speech $x(t)$, and M_{mag} is the soft magnitude mask while M_{pha} is the soft phase mask. This masking operation is inspired by the machine learning speech separation algorithm [39] and RL-based SD model [21]. Finally, the inverse STFT (ISTFT) is used to obtain the estimated (denoised) signal $\widehat{s}(t)$.

2.2. Training Scheme

During the training, each RL agent observes the state S_j, and performs an action A_j, according to its policy π. The agent interacts with the environment in multiple steps $j = \{1,2,\cdots\cdots N\}$ and in each step, it tries to obtain the maximum cumulative reward U_j given in eq. (3) as:

$$U_j = \sum_{j'=j+1}^{N} R_{j'} \tag{3}$$

Where $R_{j'}$ is the reward of a single step (step-reward) and N is the total number of steps in one training episode. During the training, there is a variation in the step-reward received in a single episode, due to randomness in the agent's choice of action and its interaction with the environment. Progress of learning is therefore estimated by averaging the reward over several episodes i.e., the cumulative reward [26]. An action's influence over the future states of the environment typically decreases over time. So future actions are discounted by a discount factor. Further discussion on the effect of the discount factor is given in the subsection 'DDPG agent'.

a) States (Observation (Obs)) Space

In our model SED_{mag}, the state S_j is the set of noisy magnitudes for all frequency bins in a single time-frame of magnitude matrix *Mag*, while it is the set of noisy phases of the phase matrix *Pha* in one time-bin for the SED_{pha} model.

b) Action Space

Fig. 2 depicts the interaction of agent and environment in our proposed model SED_{mag} model during each sample time. The sample time is defined as the time after which the agent is bound to observe the current state S_j of the environment and take action A_j (according to that state and its current policy π). In SED, the action A_j is a set of probabilities (soft masks) for all frequency bins in a single time frame. This mask is fed to the reward function, where it is required to be applied to the state S_j to estimate the step-reward produced as a result of taking that action. As

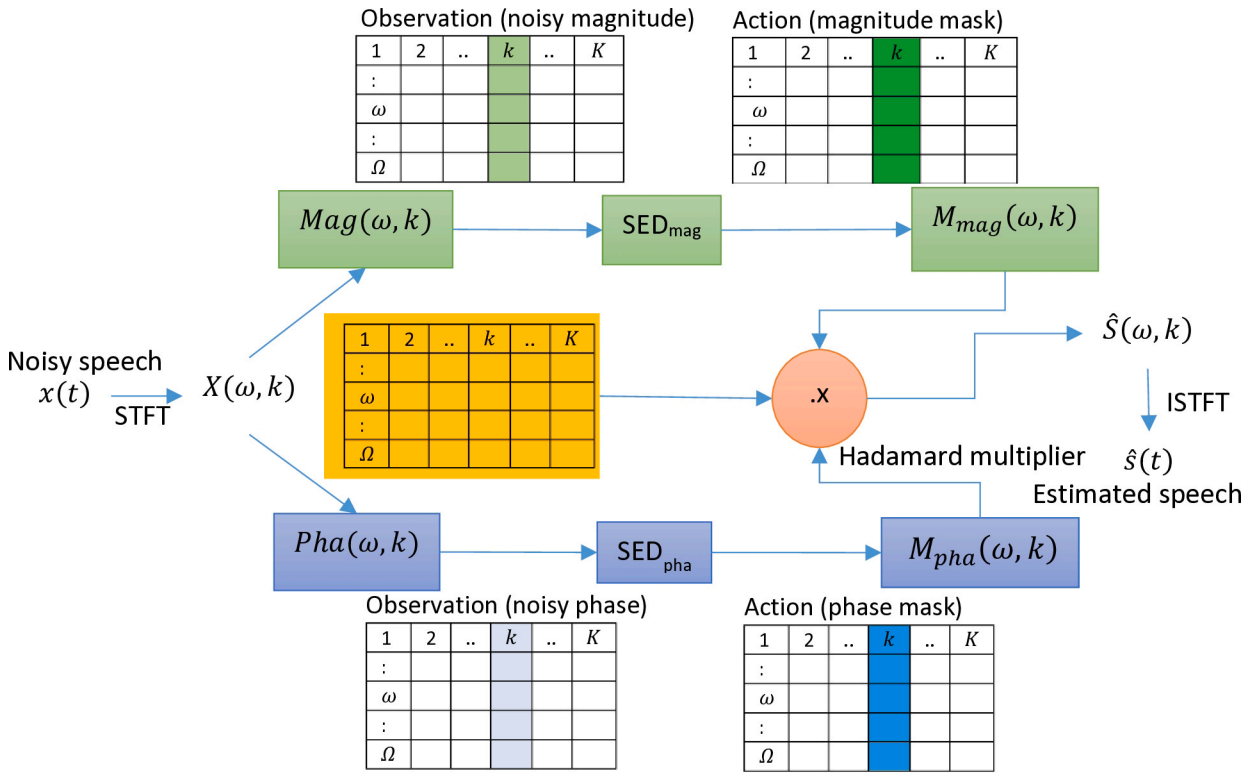


Fig. 1. Block diagram of SED during the testing phase.

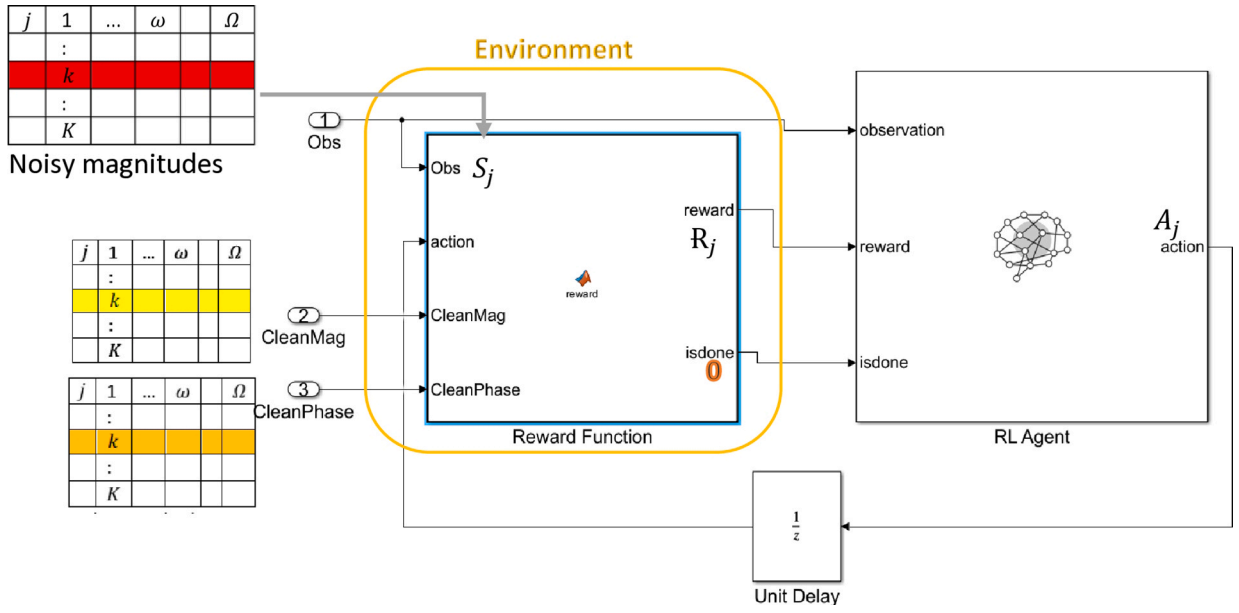


Fig. 2. Interaction of agent with environment in SED_{mag} model during each sample time. The model of SED_{pha} is similar, except that its 'Obs' signal is taken from the noisy phase matrix.

shown in Fig. 2, there is a unit delay in the feedback path (which cannot be avoided due to system's design constraints), so by the time the action A_j (produced by state S_j) is fed back to the environment, the input state has changed from the S_j to a new state S_{j+1}, which is then used for the calculation of step-reward. As there exists a naturally strong correlation between the STFT coefficients of the neighbouring time frames in the case of speech, which is enhanced further by overlapping the frames in STFT [40], this delay would not impose any problem and is not considered problematic in SED and other TF RL-based SE models (e.g. [22], and [21]).

c) Step-reward

The relative reward function from the perceptual scores is also adopted for SED from the models [22] and [25], but in a slightly different way. SED calculates the step-reward by subtracting the PESQ score of dirty speech (DIRTY) Z^{Dirty} from the PESQ score of the masked speech (MSK) Z^{MSK}.

The step-reward for the j^{th} step is given mathematically as:

$$R_j = Z^{Dirty} - Z^{MSK} \tag{4}$$

Dirty and MSK speech signals are defined differently for SED_{mag} and SED_{pha}. For SED_{mag} the dirty speech signal is defined as the speech created by combining the noisy magnitudes and the clean phases. On the other hand, the MSK speech for this model is defined as the speech obtained by applying the masks estimated by the agent, on the observations (i.e. the noisy magnitudes), and combining the product with the phases of clean speech. In the case of SED_{pha}, the dirty speech is the speech created by combining the noisy phases and clean magnitudes, while the MSK speech is obtained by applying the phase masks estimated by the agent on the observations (i.e. the noisy phases) and combining the product with the magnitudes of the clean speech.

The reward would remain negative (penalty) until the PESQ of the MSK speech generated by the RL agent is better than the PESQ score of the dirty speech. As both the MSK and the dirty speech in SED_{mag} use the same phase (phase extracted from the clean speech samples), the only way to make the reward positive is to improve the magnitude component of the MSK speech so that its PESQ score could surpass that of the dirty speech, which in turn is controlled by the mask (action) generated by the RL agent. In the case of SED_{pha}, the magnitudes of dirty and MSK speeches are similar, so the RL agent tries to improve the PESQ score of the MSK speech by cleaning its phase.

In eq. (4), the PESQ scores for the step-reward calculation are estimated for a single time frame. This differs from the model [22], which needs a minimum of five concatenated time frames of speech to calculate the PESQ scores. This is due to the difference in the PESQ calculators used for the two models. The SD model [22] uses the PESQ calculator of [23] while our proposed model uses the PESQ calculator of [24] which can calculate the score for a single time frame.

d) DDPG Agent

Both SED_{mag} and SED_{pha} use DDPG agents to estimate their soft masks. The DDPG agent is an off-policy actor-critic agent for solving problems requiring continuous action space [41]. The agent is trained off-policy using samples from the experience buffer. The experience (replay) buffer is a finite-sized cache storing the tuples (S_j, A_j, R_j, S_{j+1}) that were sampled from the environment after the agent's interaction with the environment [9]. When the replay buffer is full, the old samples are replaced [9]. At each time step, the weights of the actor and critic are updated by training them on a mini-batch of experiences, randomly selected from the experience buffer.

Directly implementing Q-learning by neural networks has proved unstable in many situations. This is avoided by using the target-actor and target-critic besides the original actor and critic networks. These target networks are replicas of their original networks in architecture but track them slowly, improving the agent's stability.

To obtain the estimated policy and the value-function, a DDPG agent maintains four function approximators. These are $\pi(S;\theta)$ for actor, $\pi_t(S;\theta_t)$ for target-actor, $Q(S,A;\boldsymbol{\varphi})$ for critic, and $Q_t = (S,A;\boldsymbol{\varphi_t})$ for target-critic, where θ, θ_t, $\boldsymbol{\varphi}$, and $\boldsymbol{\varphi_t}$ are learnable parameters of these networks respectively, Q is the Q-value critic, and π is the actor's exploration policy [41]. Both actor and critic are initialized with random weights while their target counterparts are initialized with the same values [41]. During the training step j, the actor selects and executes an action $A_j = \pi(S_j;\theta) + Noi$ for the current observation S_j, where Noi is the stochastic noise sampled from a noise process that suits the environment. It is set to the Ornstein-Uhlenbeck process for the default DDPG agent in MATLAB. Our proposed model also uses this default setting for the noise process [9]. After the execution of A_j, the step-reward R_j is calculated, the agent observes the next state S_{j+1} and stores the tuple in its experience buffer. From the buffer, a mini-batch of M random experiences is sampled. If S_{j+1} is the terminal state, the target value function y_j is set to R_j, otherwise, it is given as the sum of the step-reward R_j and the discounted future reward given as:

$$y_j = R_j + \gamma Q_t\left(S_{j+1}, \pi_t(S_{j+1};\theta_t);\boldsymbol{\varphi_t}\right) \tag{5}$$

where the discount factor γ determines how the rewards at the individual time steps are weighted. The discount factor either makes the agent long or short-sighted [26]. If a value of the '*NumStepsToLookAhead*' hyperparameter in DDPG training options is set to P, the cumulative reward is given as:

$$U_j = R_j + \gamma R_{j+1} + \gamma^2 R_{j+2} + \cdots\cdots\cdots + \gamma^L R_{j+P} \tag{6}$$

Also the target function value y_j is given as:

$$y_j = R_j + \gamma Q_t\left(S_{j+1}, \pi_t(S_{j+1};\theta_t);\boldsymbol{\varphi_t}\right) + \cdots\cdots\cdots\cdots + \gamma^L Q_t\left(S_{j+P}, \pi_t(S_{j+P};\theta_t);\boldsymbol{\varphi_t}\right) \tag{7}$$

The critic parameters are updated by minimizing the loss L across all the M samples of the mini-batch selected from the experience buffer. This loss L is given in eq. (8) as:

$$L = \frac{1}{2M}\sum_{i=1}^{M}\left(y_i - Q(S,A;\boldsymbol{\varphi})\right)^2 \tag{8}$$

Similarly, the actor parameters are updated to update the policy gradient of the actor, required to maximize the cumulative reward, as given in eq. (9):

$$\nabla_\theta \mathrm{J} = \frac{1}{M}\sum_{i=1}^{M} G_{ai} G_{\pi i} \tag{9}$$

where G_{ai} is the gradient of critic output according to the action selected by the actor network and is given as:

$$G_{ai} = \nabla_a Q\left(S_j, A;\boldsymbol{\varphi}\right), where A = \pi\left(S_j;\theta\right) \tag{10}$$

and $G_{\pi i}$ is the gradient of actor output with respect to the actor network given mathematically as:

$$G_{\pi i} = \nabla_\theta \pi\left(S_j;\theta\right) \tag{11}$$

Both gradients are updated for each observation S_j. Finally, the parameters of the replica networks i.e. the target-actor and the target-critic are updated using the smoothing mechanism given in eqs. (12) and (13):

$$\boldsymbol{\varphi_t} = \tau\boldsymbol{\varphi} - (1-\tau)\boldsymbol{\varphi_t} \tag{12}$$

$$\theta_t = \tau\theta - (1-\tau)\theta_t \tag{13}$$

where τ is the smoothing factor, which is $\ll 1$.

e) Is-done or Termination Condition

Each training episode of the RL agent consists of many steps, but they all may not be executed. The end of an episode is decided by the fulfilment of any of these three conditions, 1) the defined goal (cumulative reward) is achieved, 2) the termination condition (also called as 'is-done' condition) occurs or 3) the maximum steps N defined for an episode are completed (where each step is a single unit of an episode). Early termination occurs if either of the first two conditions (mentioned above) is met in any of the episode's steps, resulting in the ending of that episode [26].

In our proposed model, the termination condition (depicted by the 'Is-Done' signal flowing out from the reward function block in Fig. 2) is set to zero, indicating that there is no early termination condition. Also, the cumulative reward is set to a very high value to ensure that all steps of each episode are executed.

f) Reset function for Training and Testing

After an episode ends, a new episode starts with the environment states given in the reset function. In SED, after the completion of a single episode, the agent restarts the next episode by extracting a mini-batch of M experiences, starting from a random index of the experience buffer.

However, the reset function also ensures that the correct phases and magnitudes of clean speech corresponding to each noisy sample in the mini-batch are transferred from the base workspace to the environment created in Simulink, where they play an important role in the calculation of the step-reward. During testing, as the reward function is not required to be executed, the system simply takes a noisy matrix (magnitude/phase) and passes its columns sequentially to the reset testing function, which takes their transpose and concatenates the sample time in the beginning of each vector and inputs them to SED_{mag} and SED_{pha} respectively, which in turn produce magnitude and phase soft masks for each time frame at the output. These mask vectors are then stored sequentially by the main program as matrices (M_{mag}/M_{pha}) for all time frames belonging to a single clip of noisy test speech in separate MAT files. The SED algorithm is summarized in Table 1.

3. Experimental Settings

3.1. Training and Testing Datasets

The dataset used for training and testing our proposed model is VCKT-DEMAND [29] (a publicly available Valentini-Botinhao dataset), comprising parallel clean-noisy clips sampled at 48KHz. The dataset has two speech corpora, the first has utterances of 28 speakers, and the second has utterances of 56 speakers for training. For SED training and testing, the '28 speaker' dataset is used. In this dataset, the speech from 14 male and 14 female speakers is mixed with 10 different types of real and synthetic noises at four different SNR levels (15, 10, 5, and 0dB). The test data has utterances from 2 speakers mixed with noise of five types at four different SNR levels (17.5, 12.5, 7.5, and 2.5dB). The speakers as well as the noise types used for the trainset are not repeated for the test dataset. The duration of the training dataset is almost 9.5 h and that of the test set is 34 min. Our system uses only 80.85 min of training data to reduce the training duration. However, the testing of SED is carried out on the entire test dataset to make a fair comparison with the competing models. Both the SED_{mag} and SED_{pha} are trained on all four SNR levels (15, 10, 5, and 0 dB) of the training dataset and they are then tested for all SNR levels of the test dataset. As the utterances in each dataset are of variable durations, they are concatenated together to form two audio files 1) train file and 2) test file. The test file is then clipped into multiple files of equal duration (2 s each). So, it is probable that a single two-second clip may contain portions belonging to two speakers or two different SNR levels. If the concatenation before chopping is not done, much of the data would be wasted in case if the original audio clip's duration is not a multiple of two seconds. For testing the trained model on mismatched acoustic conditions, 10 samples (each of 2 s duration) are taken from the TIMIT speech corpus [42] and mixed with the white noise taken from the NOISEX-92 corpus [43]. In order to make a fair comparison with the results obtained on the Valentini-Botinhao dataset [29], the TIMIT speech samples are mixed at SNR levels of 10dB, as it is the average SNR (average of 17.5, 12.5, 7.5, and 2.5dB) of the test dataset [29]. As the test dataset of [29] has no samples at very low SNR conditions (the lowest being at an SNR level of 2.5dB), so for testing under very low SNR conditions, the data is manufactured by mixing ten clean speech samples from the TIMIT dataset with two SNR levels of 0 and −5dB.

The input data for the SED model is required to be in the form of a vector with its elements representing the noisy magnitudes (in the case of SED_{mag}) or noisy phases (in the case of SED_{pha}) of the STFT matrix for all frequencies in a single time-frame. It is observed that the larger this vector is, the more the chance that the system crashes during the training. So, to reduce the size of the input vector, the training and testing dataset is resampled at 8KHz and then its STFT is taken. The STFT parameters are listed in Table 2.

The resulting 64-point complex-valued STFT matrix is converted to

Table 2
STFT parameters.

Parameters	Values
Window Shape	Hanning
Window length	64 points
Number of discrete Fourier transform (DFT) points	64 samples
Overlap length	50% (32 samples)
Sampling frequency	8 KHz
Training data duration	80.85min
Testing data duration	1036 clips of 2s each

Table 1
SED training and testing algorithms.

SED training and testing algorithm
Training algorithm
Initialize the step counter j, RL training parameters and the experience buffer.
Initialize actor and critic networks with θ and, $\boldsymbol{\varphi}$, and set target-actor and target critic with $\theta_t = \theta$, and $\boldsymbol{\varphi_t} = \boldsymbol{\varphi}$.
For episode $1, \cdots\cdots E$ do
Initialize a random noise process *Noi* for action exploration.
Set $j = 1$, and initialize the system to S_j: the state defined in reset function.
while $j = 1, \cdots N$, and cumulative reward U_j not achieved **do**
Select action $A_j = \pi(S_j; \theta) + Noi$ according to the current policy π and exploration noise.
Execute the action A_j, observe reward R_j, and the new state S_{j+1}.
Store the tuple (S_j, A_j, R_j, S_{j+1}) in the experience buffer.
Sample a random mini-batch of size M from the experience buffer.
Estimate y_j as in eq. (5).
Update the critic by minimizing the loss L given in eq. (8).
Update the actor policy by using the sampled policy gradient $\nabla_\theta J$ given in eq. (9).
Update the target networks $\boldsymbol{\varphi_t}$ and θ_t, using eqs. (12) and (13) respectively.
end while
end for
Testing algorithm
Take a noisy speech clip $x(t)$ and obtain its Fourier transform $X(\omega, k)$.
Convert $X(\omega, k)$ from complex to phasor representation and store the magnitudes and phases in *Mag* and *Pha* matrices respectively.
for count=1: num_of_columns in X
Input $Mag(1:\Omega, count)$ to SED_{mag} and $Pha(1:\Omega, count)$ to SED_{pha}.
Output: Vectors of magnitude and phase soft masks
Concatenate the output vectors column-wise in separate matrices M_{mag} and M_{pha}.
end for
Obtain estimated speech $\widehat{S}$ from eq. (2)
Use ISFT to convert $\widehat{S}$ to $\widehat{s}(t)$.

polar form and stored as a magnitude matrix and a phase matrix in separate MAT files for the noisy speech. Similarly, the clean phases and magnitudes are also stored in separate MAT files required for the calculation of step-reward. The size of each MAT file for the training data is 64x1212570. As SED uses the DDPG agent inspired by the model [44], so its training data must be arranged in the same manner as done in the model [44]. The training STFT matrix is transposed so that each row of the transposed matrix corresponds to one STFT time frame. Each row has 64 frequency bins. The sample time is concatenated before each row, resulting in a matrix of size 1212570x65 in each training MAT file.

For each test clip of 2s, the phase and magnitude matrices are also stored in separate MAT files, and the size of each file is 64x499 according to the STFT parameters mentioned in Table 2. There is no need to transpose these matrices. Also there is no need to index each row with sample times. The reset testing function will do the job. A *'for'* loop is used to input the time frames one-by-one sequentially from each of these matrices to the trained agents, which produce soft mask vectors, which are then concatenated together to produce magnitude and phase soft mask matrices of size 64x499. These matrices are then stored in MAT files for further processing (source extraction).

3.2. Evaluation Metrics

The metrics used for the evaluation and comparison of SED with other models are PESQ [24], CSIG, CBAK, and COVL [45]. PESQ is a measure of naturalness in speech and its range is {-0.5 to 4.5}. CSIG, CBAK, and COVL scores are mean opinion scores representing signal distortion, background intrusiveness, and overall speech quality [46]

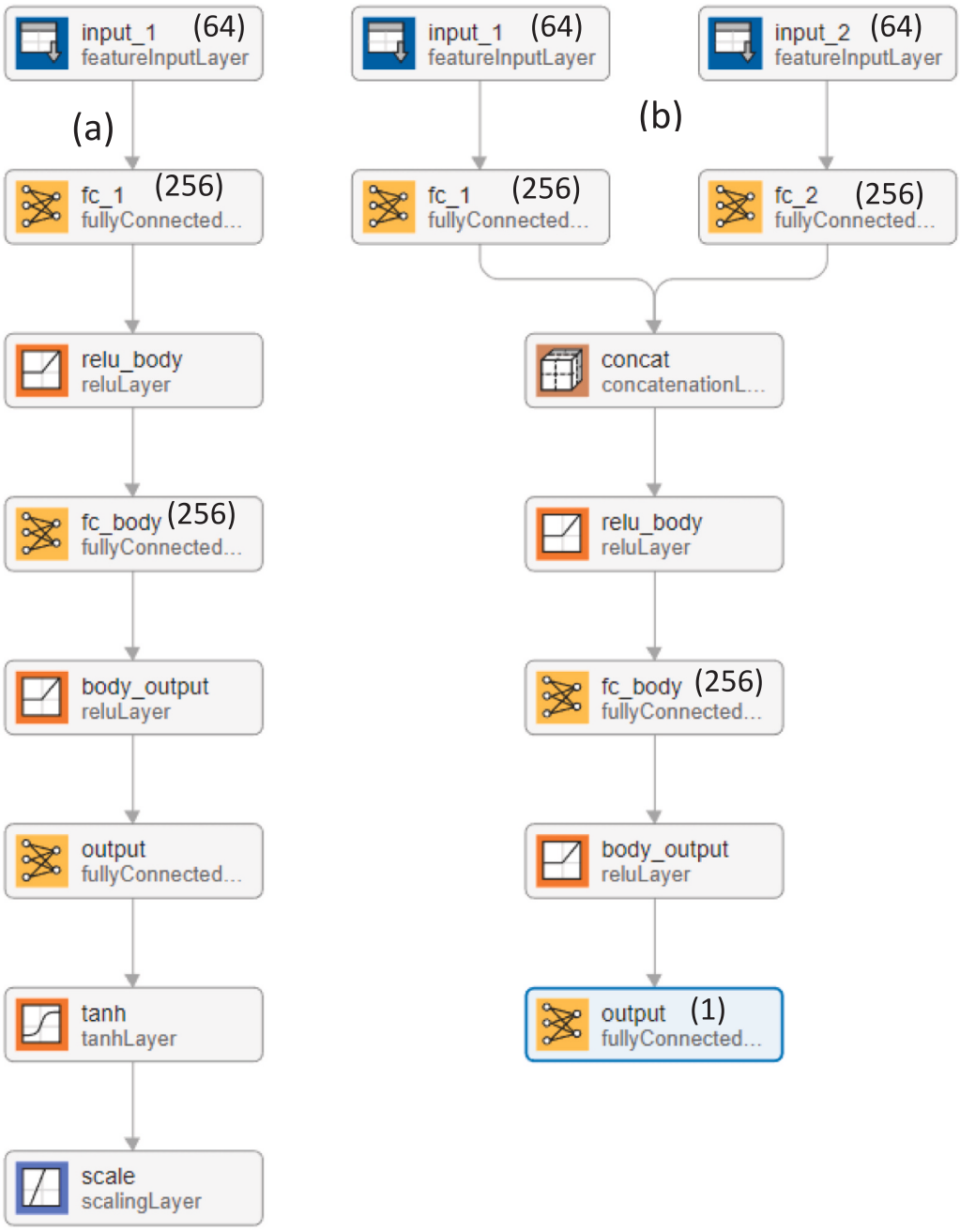


Fig. 3. Architecture of (a) actor network, (b) critic network. The number (inside the brackets) within each block shows the number of neurons in each layer.

respectively. These metrics vary in the range {1:5}. In the case of CSIG, '1′ represents 'very unnatural and very degraded' speech, while '5′ shows the signal to be 'very natural with no degradation'. The '1′ in CBAK shows the background noise to be 'very conspicuous, very intrusive', and '5′ shows it to be 'not noticeable' at all. In the case of COVL, '1′ represents the 'bad', while '5′ shows the excellent quality. These metrics are tested to correlate well with the human perception of speech quality ([45], and [47]). These metrics are selected as they are used for most of the baseline algorithms used here for SED comparison.

3.3. Actor Critic Network Architecture

The DNN architecture of actor and critic networks is shown in Fig. 3. The target-actor and critic networks are similar to their original counterparts.

The bias and scale of the scaling layer of the actor would be adjusted automatically according to the defined range of the action space. There are 98.8k learnable parameters in actor network and 164.8k in the critic network. Both state and action spaces are vectors of size 64. The upper and lower limits for the action space are 0.1 and 1, as described in the experiment section (ablation studies).

3.4. DDPG Agent Options

Important DDPG training options and their values used for both SED models are listed in Table 3. Other parameters (not shown here) are set to the default values of MATLAB.

3.5. Baseline Algorithms

Five latest state-of-the-art algorithms are used for comparing the performance of our proposed model. These models are selected as they are also trained and tested on our selected dataset [29]. A brief description of these models is given below. The results are reported directly from their respective papers.

The first model used for comparison is the SD model [25]. Like SED, it is also an RL-based system. Its working details are already given above in the 'introduction' section. The second model [48] used for comparison combines the probabilistic latent space of variational autoencoder (an unsupervised learning model) with the very high reconstruction ability of U-Net (a supervised learning model). The addition of a variational autoencoder in the bottleneck of U-Net has enabled an originally supervised learning model to adapt comfortably to the variations in environmental acoustic conditions and noise types. The third SD model used for comparison is a mapping-based SL model [49] using pix2pix GAN concatenated with a regression neural network for mapping the noisy speech's colored spectrograms to the clean speech. The fourth SD model [50] is an SSL model that proposes a novel scheme for fine-tuning to smoothly transition the model from the upstream task to the downstream task. Finally, the model [51] is a diffusion-based model using an improved stochastic differential equation. It is categorized as the USL model.

In addition to the comparisons mentioned above, the proposed model is cascaded with [49] to determine the gain in output. This application is similar to the model [22,30,32], and [15], where RL is used for improvement of already trained SE algorithms for ASR and SD applications.

Table 3
DDPG training options.

	Parameter	Value
DDPG options	Mini batch size	128
	Experience buffer length	1e+06
	Sample time	0.016
Actor and critic optimizer options	Optimizer	Adam
	Learn rate	5e-5
	Gradient threshold	10
RL training options	Maximum episodes	1000
	Maximum steps per episode	50
	Score averaging window length	5
	Stop training criteria	Average reward
	Stop training value	200

4. Results and Discussion

4.1. Ablation Studies

Initially the tests were carried on 50 clips (2s each) of test data to find the type of masking, range of action space, the suitable agent, and total episodes required for training. After figuring out the most suitable values for these parameters, the model is tested for the entire dataset [29]. The bold-faced results show the best performers in each study.

a) Type of Masking

In this study, the effect of magnitude and phase masks on system's performance is compared. First, the magnitude and phase masks are applied individually on the complex STFT matrix of noisy speech, then both masks are simultaneously applied by taking their product, as shown in eq. (2). The results are shown in Table 4.

The results of Table 4 indicate that although the "product" and the "phase-only" masks have shown significant improvement for all metrics over the unprocessed data, the "magnitude-only" mask has produced the best results. Using a single mask for the retrieval of denoised speech would save the time required to train $SED_{pha,}$ as the magnitude-only mask is enough to produce better-quality speech.

b) Range of Action Space

Ratio (soft) masks are better than binary (hard) masks for improving the quality of speech [52]. Ideally, the range of soft masks must be {0:1}. However, the results show that the output sound quality is not good in that case. So, various ranges for the action space are tested to find the most suitable limits. The results are shown in Table 5 for SED_{mag} trained for 200 episodes and tested by magnitude-only mask.

The results of different limits chosen for the action space show that as compared to other settings, the range of {0.1:1} produces higher quality speech. So, in the experiments that follow next, this setting is used.

c) Suitable Agent

As the nature of the mask required for the SD problem needs to be continuous, so in this experiment two continuous agents are tested: 1) DDPG, and 2) twin-delayed deep deterministic (TD3) policy gradient algorithm [53]. TD3 is an extension of DDPG with two critics designed to avoid overestimation of value function by a single critic, which can lead to suboptimal policy. The settings of both agents are kept similar. The results for 200 episodes, with action space limits set to {0.1:1}, and tested by the "magnitude-only" mask are shown in Table 6.

It is observed in Table 6 that with similar training parameters, the DDPG agent generates better results than the TD3 agent. However, tuning TD3 may provide better results for SE applications and needs to be explored in future.

d) Total Training Episodes

In this experiment the number of training episodes is varied and the system's performance is observed. The results of varying the number of episodes in the range {1000:1000:3000} are listed in Table 7. "Magnitude-only mask" is used for each case and the action space is {0.1:1}.

Table 4
Effect of changing the type of mask.

Type of mask	Episodes	PESQ ↑	CSIG ↑	CBAK ↑	COVL ↑
Magnitude-only	1000	**3.02**	**4.26**	**3.02**	**3.64**
Phase-only	1000	2.88	4.16	2.93	3.51
Product mask	1000	2.89	4.1	2.87	3.5
Unprocessed	––	1.4	3.07	2.2	2.21

Table 5
Effect of changing the limits of the action space.

Range	PESQ ↑	CSIG ↑	CBAK ↑	COVL ↑
{0: 1}	2.95	4.24	2.97	3.6
{0.1: 1}	**3.14**	**4.35**	**3.12**	**3.76**
{0.2: 1}	2.93	4.18	2.94	3.55
{0.1: 0.9}	2.94	4.20	2.89	3.56

Table 6
Effect of different agents.

Agent	PESQ ↑	CSIG ↑	CBAK ↑	COVL ↑
DDPG	**3.14**	**4.35**	**3.12**	**3.76**
TD3	2.55	3.90	2.75	3.20

Table 7
Effect of changing the number of episodes.

Episode	PESQ	CSIG	CBAK	COVL
1000	**3.02**	4.06	3.02	**3.65**
2000	2.93	4.2	**3.03**	3.56
3000	3.01	**4.23**	**3.03**	3.63

There is a small drop in PESQ and COVL when the training episodes are increased from 1000 to 2000 while improving CSIG slightly. However, the performance of the system at 3000 episodes is almost similar to its performance at 1000 episodes. So, in order to save the training time, the number of training episodes is set to 1000 in future experiments.

The framework the proposed algorithm after the ablation studies is depicted in Fig. 4.

4.2. Comparison with Baseline Algorithms

Table 8 shows the comparative performance of baseline algorithms with SED. Here SED is tested for the entire test dataset given in the'28 speakers' corpus of [29] after training it over 1000 episodes with "magnitude-only" mask and action space limits set to {0.1:1}.

The results show that as a stand-alone system, SED has produced the highest PESQ as compared to other systems. However, its CSIG, CBAK, and COVL are slightly lower than [50], the best performer in terms of these metrics. SED has achieved this performance while being trained on a dataset that is seven times smaller than the one used for training other models. Also, its learnable parameters are the lowest among all SD systems compared here (almost seven times lower than the second best performer [25]), making it an ideal choice for hand-held computing devices. The spectrograms of clean, noisy, and estimated speech from SED model (as a stand-alone SD system) are shown in Fig. 5.

When cascaded with the pretrained model [49], the PESQ is further improved by 0.07 points. Also, in this case, the CSIG and COVL values become equal to the model [50], while CBAK is only 0.3 points less than the model [50]. The SD model [49] is trained on spectrograms produced from speech sampled at 16kHz. It uses the noisy phase for speech reconstruction. The SED model (phase-only) cannot be used for the phase cleaning of [49] due to differences in the sampling rate of the two models. So, it is applied to the estimated speech of the model [49] after its resampling to 8kHz. In the future, retraining [49] with spectrograms of 8kHz sampled speech and combining its estimated magnitudes with the phase-only masks of SED may improve its performance even further. The resulting spectrograms of SED integration with [49], for a noisy speech clip, are depicted in Fig. 6.

Table 9 shows the improvement achieved when different types of RL agents are incorporated with other DNN SE algorithms. The direct comparison of these integrations with our cascaded network (SED + [49]) is not possible due to the differences in their datasets and the types of noises used for training and testing. So, the integrated systems are compared (under the average SNR level of 5 dB) on the basis of the gain in PESQ (Δ PESQ) and the number of episodes required to achieve it.

The results in Table 9 show that a continuous RL agent (DDPG) in SED has not only resulted in more improvement in speech quality (indicated by the highest Δ PESQ) but also has reduced the required number of episodes by 50 times, as compared to the discrete RL agents used by other models. This finding further asserts the results of [25], which also uses a continuous RL agent for its SD model. Although the results of SED with the TD3 agent shown in Table 6 are not good for the current system settings, further experiments may prove its usefulness in SD or other SE-related applications.

4.3. Testing on Mismatched Conditions

In this case, the proposed model is tested on the unseen speech corpus, noise type, and low values of SNR to see its generalization

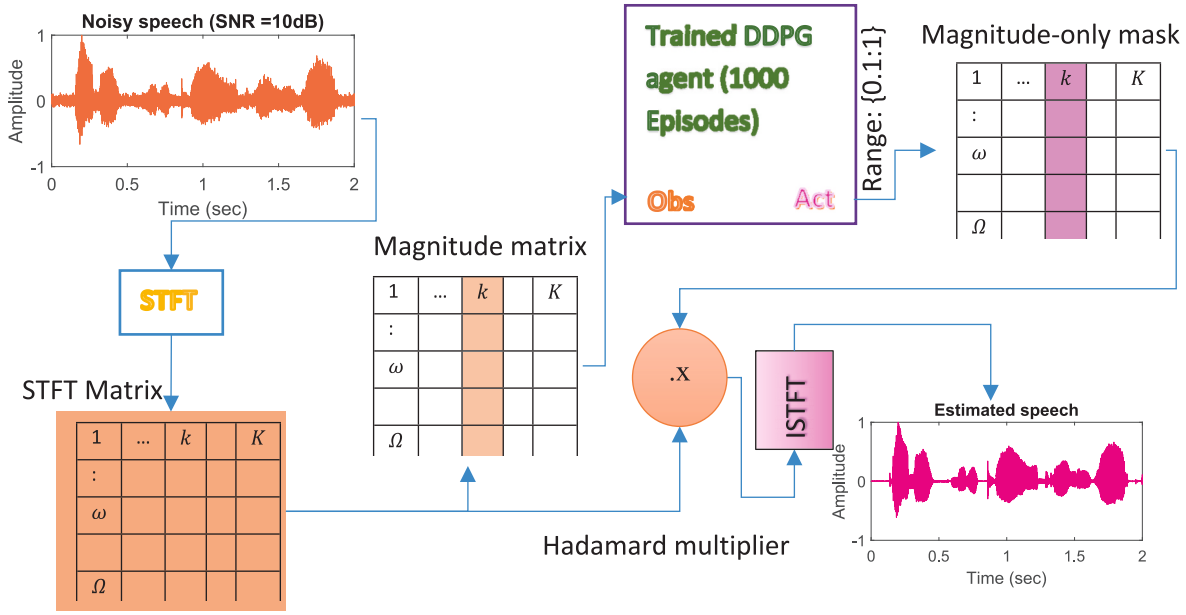


Fig. 4. Framework of the proposed model after ablation studies.

Table 8
Comparison to baseline models.

Baseline algorithms	DL class	PESQ ↑	CSIG ↑	CBAK ↑	COVL ↑	Dataset used for training	Network parameters (M)
Unprocessed	———	1.4	3.07	2.2	2.21	––	––
SED (Proposed)	RL	**2.95**	4.10	2.94	3.53	**1/7**	**0.26**
[25]	RL	2.360	3.4	3.0	2.87	Full	3.22
[49]	SL	2.24	3.65	2.69	2.97	Full	64.98
[48] (DVU-Net (Ma-Only)	USL N/W within an SL N/W	2.45	NA*	NA*	NA*	Full	25.8
[51]	USL	2.9	NA*	NA*	NA*	Full	34.3
[50]	SSL	2.86	**4.3**	**3.3**	3.6	Full	95
Cascaded system ([49] + SED)	SL +RL	**3.02**	**4.3**	3.0	**3.61**	Pretrained	65.24

NA* = not available, N/W = network.

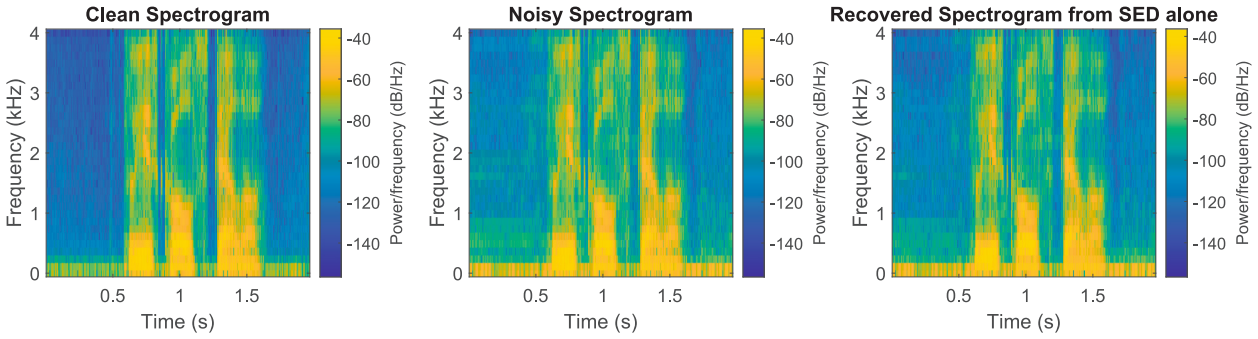


Fig. 5. Clean, Noisy and estimated speech spectrograms from SED model.

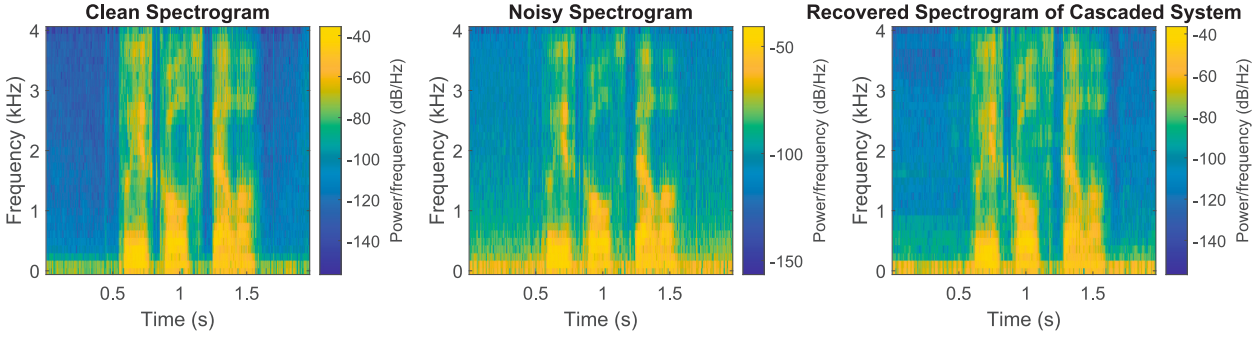


Fig. 6. Clean, noisy and recovered spectrograms of cascaded system.

Table 9
Performance improvement for different trained models by RL agents.

System integrated with RL	RL model used for improvement	RL-agent incorporated	Δ PESQ ↑	RL training episodes
[49]	SED	**DDPG**	**0.78**	**1000**
DNN mapping	[22]	Q-Learning	0.2	50,000
[54]	[30]	DQN	0.02	NA*
ASR	[32]	Q-learning	0.11	NA*

NA* = not available.

ability. For mismatched and low SNR conditions, the dataset is prepared according to the recipe given in the subsection 'training and testing datasets'.

The results under mismatched dataset and low SNR conditions are shown in Table 10.

As clear from Table 10, the system's performance drops significantly under the mismatched conditions. The exact reason needs proper investigation; it seems that the smaller network size of the DDPG agent is the root cause responsible for its reduced generalization ability under unseen conditions. In the future, using convolutional or recurrent layers

Table 10
Testing on mismatched conditions.

Speech corpus	Av. SNR (dB)	PESQ ↑	CSIG ↑	CBAK ↑	COVL ↑
VCKT-DEMAND	**10**	**2.95**	**4.10**	**2.94**	**3.53**
TIMIT	10	1.81	3.02	1.60	2.32
TIMIT	0	1.34	2.00	1.20	1.56
TIMIT	−5	1.23	1.60	1.11	1.27

or designing a deeper network may prove beneficial for improving its generalization ability, as deeper networks usually enhance the generalization of feedforward neural networks [55].

5. Limitations

Although the training duration of SED is not lengthy due to requiring fewer episodes and using GPU, the testing time is not suitable for online applications. Processing a two-second audio clip from a trained model requires almost one minute on a 12th Gen Intel(R) Core(TM) i5-12450H, 2.00 GHz processor. Reducing this time is the main challenge. Secondly, working with very high sampling rates of 16, 32, or 44.1kHz results in large sizes of state and action vectors, eventually causing the system to

crash. Thirdly, the system SED_{pha} does not yield significant improvement compared to that achieved by the "magnitude-only" network. This limited success stems from the inherent random characteristics exhibited by the phase spectra in the TF domain [48]. As in the model [56], providing the contextual magnitude information to the phase-processing SED may improve its performance. Finally, the system does not generalize well to the mismatched acoustic conditions.

6. Conclusion

In this paper, an RL-based SD algorithm is proposed which uses DDPG; an off-policy continuous action space agent. On the noisy speech dataset [29], our proposed model has generated speech quality on par with the other state-of-the-art algorithms, requiring far lesser training data and the number of learnable parameters than required by the SD algorithms based on SL, USL, or SSL. Also, as compared to the discrete agents, the continuous agent used in our proposed model has provided more gain in quality achieved in much fewer episodes. However, it does not generalize well to the unseen acoustic conditions. Exploring more continuous agents and algorithms for reducing the testing time, and tweaking the network's size and architecture for SD and general SE applications may prove beneficial in future research

CRediT authorship contribution statement

Sania Gul: Software, Investigation, Data curation, Writing – original draft, Validation, Methodology, Writing – review & editing, Visualization, Formal analysis, Conceptualization. **Muhammad Salman Khan:** Resources, Investigation, Conceptualization, Formal analysis, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data Availability

The Valentini-Botinhao dataset used during the current study is publicly available at repository, https://doi.org/10.7488/ds/2117.

References

[1] Li Y, Jin X, Tong L, Zhang LM, Yao YQ, Yan H. A speech enhancement model based on noise component decomposition: inspired by human cognitive behavior. Appl Acoust 2024 May;15(221):109997.
[2] Gul S, Khan MS. A survey of audio enhancement algorithms for music, speech, bioacoustics, biomedical, industrial and environmental sounds by image U-Net. IEEE Access 2023.
[3] Purwins H, Li B, Virtanen T, Schlüter J, Chang SY, Sainath T. Deep learning for audio signal processing. IEEE J Sel Top Signal Process 2019 Mar 31;13(2):206–19.
[4] Chinnamgari SK. R Machine Learning Projects: Implement supervised, unsupervised, and reinforcement learning techniques using R 3.5. Packt Publishing Ltd 2019. Jan 14.
[5] Liu L, Guan H, Ma J, Dai W, Wang G, Ding S. A mask free neural network for monaural speech enhancement. arXiv preprint arXiv:2306.04286. 2023 Jun 7.
[6] Bie X, Leglaive S, Alameda-Pineda X, Girin L. Unsupervised speech enhancement using dynamical variational autoencoders. IEEE/ACM Trans Audio Speech Lang Process 2022 Sep;16(30):2993–3007.
[7] Hao X, Xu C, Xie L. Neural speech enhancement with unsupervised pre-training and mixture training. Neural Netw 2023 Jan;1(158):216–27.
[8] Ayilo JE, Sadeghi M, Serizel R. Diffusion-based speech enhancement with a weighted generative-supervised learning loss. InICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2024 Apr 14 (pp. 12506-12510). IEEE.
[9] Lillicrap, Timothy P., Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous Control with Deep Reinforcement Learning. ArXiv:1509.02971 [Cs, Stat], September 9, 2015. https://arxiv.org/abs/1509.02971.
[10] Wang YC, Venkataramani S, Smaragdis P. Self-supervised learning for speech enhancement. arXiv preprint arXiv:2006.10388. 2020 Jun 18.
[11] Liu S, Mallol-Ragolta A, Parada-Cabaleiro E, Qian K, Jing X, Kathan A, et al. Audio self-supervised learning: A survey Patterns 2022 Dec 9;3:12.
[12] Huang Z, Watanabe S, Yang SW, García P, Khudanpur S. Investigating self-supervised learning for speech enhancement and separation. InICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2022 May 23 (pp. 6837-6841). IEEE.
[13] Arulkumaran K, Deisenroth MP, Brundage M, Bharath AA. Deep reinforcement learning: a brief survey. IEEE Signal Process Mag 2017 Nov 9;34(6):26–38.
[14] Qiang W, Zhongli Z. Reinforcement learning model, algorithms and its application. In2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC) 2011 Aug 19 (pp. 1143-1146). IEEE.
[15] Latif S, Cuayáhuitl H, Pervez F, Shamshad F, Ali HS, Cambria E. A survey on deep reinforcement learning for audio-based applications. Artif Intell Rev 2023 Mar;56 (3):2193–240.
[16] Prudencio RF, Maximo MR, Colombini EL. A survey on offline reinforcement learning: Taxonomy, review, and open problems. IEEE Trans Neural Networks Learn Syst 2023 Mar 22.
[17] Zheng H, Luo X, Wei P, Song X, Li D, Jiang J. Adaptive policy learning for offline-to-online reinforcement learning. InProceedings of the AAAI Conference on Artificial Intelligence 2023 Jun 26 (Vol. 37, No. 9, pp. 11372-11380).
[18] Reinforcement Learning Agents: https://www.mathworks.com/help/reinforcement-learning/ug/create-agents-for-reinforcement-learning.html.
[19] Sutton RS. Reinforcement learning: an introduction. Bradford Book 2018.
[20] Finn C, Abbeel P, Levine S. Model-agnostic meta-learning for fast adaptation of deep networks. InInternational conference on machine learning 2017 Jul 17 (pp. 1126-1135). PMLR.
[21] Zhou W, Ji R, Lai J. MetaRL-SE: a few-shot speech enhancement method based on meta-reinforcement learning. Multimed Tools Appl 2023 Nov;82(28):43903–22.
[22] Koizumi Y, Niwa K, Hioka Y, Kobayashi K, Haneda Y. DNN-based source enhancement self-optimized by reinforcement learning using sound quality measurements. In2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2017 Mar 5 (pp. 81-85). IEEE.
[23] Recommendation IT. Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs. Rec. ITU-T P. 862. 2001.
[24] ITU (2007). Wideband extension to Recommendation P.862 for the assessment of wideband telephone networks and speech codecs. ITU-T Recommendation P.862.2.
[25] Hao X, Xu C, Xie L, Li H. Optimizing the perceptual quality of time domain speech enhancement with reinforcement learning. Tsinghua Sci Technol 2022 Jun 21;27 (6):939–47.
[26] Reinforcement Learning Onramp | Self-paced online courses – MATLAB. matlabacademy.mathworks.com, https://matlabacademy.mathworks.com/details/reinforcement-learning-onramp/reinforcementlearning (accessed Jun. 7, 2023).
[27] Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Van Den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S. Mastering the game of Go with deep neural networks and tree search. nature. 2016 Jan;529(7587):484-9.
[28] Seide F, Li G, Chen X, Yu D. Feature engineering in context-dependent deep neural networks for conversational speech transcription. In2011 IEEE Workshop on Automatic Speech Recognition & Understanding 2011 Dec 11 (pp. 24-29). IEEE.
[29] Valentini C. Noisy speech database for training speech enhancement algorithms and TTS Models. University of Edinburgh. School of Informatics. Centre for Speech Research. 2016. Accessed: Dec. 20, 2023. [Online] Available: https://datashare.is.ed.ac.uk/handle/10283/2791.
[30] Fakoor R, He X, Tashev I, Zarar S. Reinforcement learning to adapt speech enhancement to instantaneous input signal quality. arXiv preprint arXiv:1711.10791. 2017 Nov 29.
[31] Alamdari N, Lobarinas E, Kehtarnavaz N (2020) Personalization of hearing aid compression by human in-the-loop deep reinforcement learning. IEEE Access 8: 203503–203515. https:// doi. org/ 10. 1109/ACCESS. 2020. 30357 28.
[32] Shen YL, Huang CY, Wang SS, Tsao Y, Wang HM, Chi TS. Reinforcement learning based speech enhancement for robust speech recognition. InICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2019 May 12 (pp. 6750-6754). IEEE.
[33] Ryu S, Lim J, Lee YS. Narrowband active Noise Control with DDPG based on Reinforcement Learning. Int J Automot Technol 2024 Aug;23:1–9.
[34] Kuo SM, Morgan DR. Active noise control: a tutorial review. Proc IEEE 1999 Jun;87 (6):943–73.
[35] Nossier SA, Wall J, Moniri M, Glackin C, Cannings N. Mapping and masking targets comparison using different deep learning based speech enhancement architectures. In2020 international joint conference on neural networks (IJCNN) 2020 Jul 19 (pp. 1-8). IEEE.
[36] Wang Y, Narayanan A, Wang D. On training targets for supervised speech separation. IEEE/ACM Trans Audio Speech Lang Process 2014 Aug 28;22(12): 1849–58.
[37] Odelowo BO, Anderson DV. A study of training targets for deep neural network-based speech enhancement using noise prediction. In2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2018 Apr 15 (pp. 5409-5413). IEEE.
[38] Zhao Y, Wang D, Merks I, Zhang T. DNN-based enhancement of noisy and reverberant speech. In2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2016 Mar 20 (pp. 6525-6529). IEEE.
[39] Mandel MI, Weiss RJ, Ellis DP. Model-based expectation-maximization source separation and localization. IEEE Trans Audio Speech Lang Process 2009 Aug 11; 18(2):382–94.

[40] Benesty J, Chen J, Habets EA. Speech enhancement in the STFT domain. Springer Science & Business Media; 2011 Sep 18.
[41] Deep Deterministic Policy Gradient (DDPG) Agent. https://www.mathworks.com/help/reinforcement-learning/ug/ddpg-agents.html.
[42] Dapra timit, acoustic phonetic continuous speech corpus 2019 accessed 10th January.
[43] NOISEX-92 dataset. GitHub - speechdnn/Noises: Noise15 , Noisex-92 and Nonspeech.
[44] Du Y, Li F, Kurte K, Munk J, Zandi H. Demonstration of intelligent HVAC load management with deep reinforcement learning: real-world experience of machine learning in demand control. IEEE Power Energ Mag 2022 Apr 21;20(3):42–53.
[45] Hu Y, Loizou PC. Evaluation of objective quality measures for speech enhancement. IEEE Trans Audio Speech Lang Process 2007 Dec 18;16(1):229–38.
[46] Kim E, Seo H. SE-Conformer: Time-Domain Speech Enhancement Using Conformer. In Interspeech 2021 Aug 30 (pp. 2736-2740).
[47] Gul S, Khan MS, Yoma NB, Shah SW. Enhancing the correlation between the quality and intelligibility objective metrics with the subjective scores by shallow feed forward neural network for time–frequency masking speech separation algorithms. Appl Acoust 2022 Jan;1(188):108539.
[48] Nustede EJ, Anemüller J. On the Generalization Ability of Complex-valued Variational U-Networks for Single-Channel Speech Enhancement. IEEE/ACM Transactions on Audio: Speech, and Language Processing; 2024 Aug 15.
[49] Gul S, Khan MS, Fazeel M. Single channel speech enhancement by colored spectrograms. Comput Speech Lang 2024;101626:Feb.
[50] Yang H, Kang HG. On Fine-Tuning Pre-Trained Speech Models With EMA-Target Self-Supervised Loss. In ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2024 Apr 14 (pp. 6360-6364). IEEE.
[51] Liu S, Jiang Y, Wu Z, Yao L, Yang Q. Drift-DiffuSE: Diffusion Model with Learnable Drift Term for Speech Enhancement. In 2024 International Joint Conference on Neural Networks (IJCNN) 2024 Jun 30 (pp. 1-8). IEEE.
[52] Chen P, Nguyen BT, Iwai K, Nishiura T. Threshold-based Combination of Ideal Binary Mask and Ideal Ratio Mask for Single-Channel Speech Separation. Information 2024 Oct 4;15(10):608.
[53] Twin-Delayed Deep Deterministic (TD3) Policy Gradient Agent: https://www.mathworks.com/help/reinforcement-learning/ug/td3-agents.html.
[54] Tashev I, Lovitt A, Acero A. Unified framework for single channel speech enhancement. In: In 2009 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing; 2009. p. 883–8.
[55] Franco L, Jerez JM, Bravo JM. Role of function complexity and network size in the generalization ability of feedforward networks. In International Work-Conference on Artificial Neural Networks 2005 Jun 8 (pp. 1-8). Berlin, Heidelberg: Springer Berlin Heidelberg.
[56] Yin D, Luo C, Xiong Z, Zeng W. Phasen: A phase-and-harmonics-aware speech enhancement network. In Proceedings of the AAAI Conference on Artificial Intelligence 2020 Apr 3 (Vol. 34, No. 05, pp. 9458-9465).