



Speech synthesis using generative adversarial network for improving readability of Hindi words to recuperate from dyslexia

Geeta Atkar¹ · Priyadarshini Jayaraju¹

Received: 1 September 2020 / Accepted: 5 January 2021 / Published online: 15 February 2021
© Springer-Verlag London Ltd., part of Springer Nature 2021

Abstract

Children learn and develop their abilities at their own pace. One of the most basic skills that they acquire is reading. However, some children struggle with reading longer than their friends, and in such a case, it is possible that they have a learning disorder known as dyslexia. The paper aims to use neural networks, namely generative neural networks, for generating raw audio data of two- or three-letter Hindi words. Using the generated data, a system will be built that will pronounce generated words for children recuperating from dyslexia. The system aims to be an effective helping tool for teachers to speed up the recuperation process by making the child repeat the correct pronunciation of the word. The system uses advance Mel-generative adversarial network neural network for working with Mel-spectrograms of the raw audio, by which the system will model its own audio iteratively, until a satisfactory result is achieved. Generated audio sample contains the Hindi words which will be taught to children. Mel-generative adversarial network will be used to generate audio samples since it provides better results compared to other existing models. 300 basic two- or three-letter Hindi words are taken as an input for assisting 5- to 8-year children. Minimum opinion score is calculated for comparison.

Keywords Dyslexia · Generative adversarial network · Minimum opinion score · MelGAN · WaveGAN

1 Introduction

Dyslexia is nothing but learning disorder occurring in children. Learning disabilities include phonological dyslexia, surface dyslexia, rapid automatic naming dyslexia, double deficit dyslexia, dyscalculia, dysgraphia, left right confusion. This paper focuses on reading disability for children age in between 5 and 8 years. Here, focus is on learning disability in Hindi language of two- or three-letter word. It hampers the child's ability to understand, learn and manipulate the sounds of certain or all alphabets or words as a whole. It becomes very difficult for dyslexic people to learn new words or break big words into small manageable chunks they can sound out. This may cause the child to have difficulty in reading, writing and

spelling. Contrary to the popular believe, dyslexia is not a reflection of how intelligent a child is—in fact, it is termed as a gap between one's ability and achievement. There are several strategies which can be used for compensating the weakness in decoding sounds. However, the most successful one is speech therapy. Speech therapists believe that understanding sounds and the lip formation while those sounds are uttered matters the most in closing the learning gap. Dyslexic children in care of good therapists are often told to keep their hands on the therapist's lips while he utters the word so that the child can learn the pronunciation as well as lip formation of the word being uttered. By estimation, it is believed that as many as one in five children has dyslexia. Diagnosing dyslexia at a young age is very important so that a proper training method is used.

✉ Geeta Atkar
geeta.atkar2016@vitstudent.ac.in

Priyadarshini Jayaraju
Priyadarshini.j@vit.ac.in

¹ School of Computer Science and Engineering, Vellore Institute of Technology, Chennai 600127, India

1.1 Helping dyslexic children

One of the most used methods is teaching the child the correct pronunciation by breaking the word in syllables and then taking it forward syllable by syllable. It is almost

always accompanied with the physical interaction of a speech therapist and the dyslexic child. However, a reinforcement approach, such as making the child repeat the correct pronunciation again and again, is also effective. It is very important the child is not burdened with learning big words straight away. The process needs to take a step-by-step approach. It is also shown, through research, that it is much easier for a kid to learn words in their mother tongue.

2 Literature survey

Generative adversarial networks [1] were devised by Ian Goodfellow for the purpose of image generation by using a unsupervised strategy for mapping low-dimensional latent vectors to high-dimensional data. This framework is used for estimating generative models via an adversarial process, in which two models were simultaneously trained: a generative model G that captures the data, and a discriminative model D that is used to estimate the probability that a sample came from the training data rather than G itself. The procedure for training G is to maximize the probability of D making any mistake. This framework is similar to a minimax two-player game. In the use case where G and D are defined as multilayer perceptron, the entire system can be trained with back-propagation [2]. Qualitative and quantitative evaluation of the generated samples was done using experimental procedures. However, this whole research was for image generation, not for audio synthesis. It is important to note that it is the basis of the upcoming studies. One of the first attempts to synthesize audio using GAN was done during the adversarial audio synthesis [3] research. Generally, sampling of audio signals is done at high resolutions, and learning to synthesize raw audio requires getting the skeletal structure across a range of different time frames. By that time, generative adversarial networks (GANs) had been successful at generating images that were coherent for long time now, but they were not used for audio generation [4]. In this paper, researchers introduced WaveGAN, which was the primary attempt at using GANs for unsupervised synthesis of raw audio by using a waveform structure. It ended up synthesizing one second long sample of audio waveforms which could only be primarily used for generating sound effects. However, there research demonstrated that it could produce understandable words using a very tiny dataset of English vocabulary. WaveGAN could also synthesize audio using other domains like bird sounds, piano notes, etc. This research while beneficial in moving our study further falls short at the one second frame requirement. We need Hindi words which might take longer than a single second. The first research to use deep learning for

audio generation was done by a group of people working on the Google Deep Mind project. They introduced WaveNet [5], a deep neural network for making new raw audio samples using waveforms [6]. The model is completely probabilistic and autoregressive. It was trained on data which had thousands of samples per second of raw audio. They also applied it to TTS which yielded state-of-the-art performance, where human listeners rated it as more human sounding than the previous systems. A WaveNet can capture the features of several speakers with equal and good fidelity, and could switch between them easily. It was also trained on music notes and was found out that the notes generated were of high quality. This was an ideal research for the study, but the fidelity achieved was not at par with requirements [7]. Dyslexic children require the speech to be clear and crisp. Following WaveNet, a vocoder was built using the same. It learned directly from speech waveform samples and achieved very high-quality synthetic speech. However, for training the acoustic model, significant degradation in quality was shown. To overcome the issue, a conditional GAN was incorporated into multi-speaker speech synthesis. Wasserstein GAN [8] with gradient penalty uses mean-squared error and adversarial losses as parts of objective functions. Experimental results show that models trained using the WGAN-GP framework using back-propagation achieved the highest evaluation scores in terms of quality and speaker similarity. One of the major applications of audio synthesis with the help of GAN was voice conversion (VC). Applied to musical instruments first, VC converted notes from a guitar to a piano automatically. Such an approach opened ways for researchers to convert voices of people such as gender change, etc. One of the recent VC models that uses GAN was proposed by as a parallel-data-free voice-conversion (VC) method. It could master to map from the source to target articulation without using the parallel data. Their model, called CycleGAN-VC [9], used a cycle-consistent adversarial network (CycleGAN) with gated convolutional neural networks (CNNs). It also used an identity-mapping loss. They configured a CycleGAN with gated CNNs and trained it with an identity-mapping loss. An objective evaluation was done which showed that the converted feature series was very natural in terms of modulation. The search for a superior performance for a VC algorithm was still on. Even with their remarkable success in development of machine learning, when a nonparallel data source was used, their performance was not up to the par. Researchers proposed a cycle consistent adversarial network (CycleGAN) for nonparallel data-based VC training [10]. Originally, CycleGAN was developed for image-to-image conversion domain, but with a few tweaks, it was able to be applied

on audio as well [4]. This was one of the first searches that showed that the performance of a nonparallel VC method can exceed the parallel VC methods. In terms of speech synthesis, the researchers now gained momentum with WaveNet and WaveGAN exploration. To further advance the domain of audio modelling, Deep Mind researchers proposed a model that used GAN for text to speech, named GAN-TTS [11]. Their architecture was composed of a conditional feed-forward generator which produced raw speech audio, and an ensemble of discriminators which operated on random windows of different sizes. The discriminator is so particular that it analysed the audio in terms of how real it feels generally as well as how it corresponds to the utterance, that is, its pronunciation. To measure their performance, they employed both subjective human evaluation (MOS—mean opinion score) [12], as well as quantitative metrics (Fréchet Deep Speech Distance and Kernel Deep Speech Distance), which were found to be well correlated with MOS. They were able to show that GAN-TTS was capable of generating high-fidelity audio speech with naturalness comparable to the other state-of-the-art models, and efficient feedforward generator made is highly parallel. Researchers at Lyrebird AI showed that it was possible to train GANs reliably to generate good quality waveforms by changing architectural structure and refining the training methods. Their subjective evaluation metric used was MOS, and it showed the efficacy of the approach. They introduced MelGAN [13], a non-autoregressive feedforward convolutional architecture which performed in an audio waveform generation set-up with the help of GAN. This was the first model that trained GAN without any extra distillation or perceptual loss functions while still outputting high-fidelity results for text to speech synthesis model. They used Mel-spectrogram inversions as a feed to the GAN set-up of discriminator and generator. There model was completely no autoregressive, fully convolutional and had significantly fewer parameters [14]. This made it possible to run the model on smaller system architecture, and their PyTorch implementation ran much faster than in real time on a GTX 1080Ti GPU and more than $3 \times$ faster than in real time on CPU. This model is the basis of this study, and MelGAN is used for text to speech synthesis in this study [15]. GANs have great progress in domain translation for images [4], and video-to-video synthesis [16], but for audio, not many models are made.

3 Proposed system

The proposed system teaches dyslexic children the correct pronunciation of a two- or three-letter Hindi word through a reinforcement approach. The child will be asked to repeat the word until he is able to pronounce correctly. Repetitive mistakes will lead to segregation of the word into its syllables, which would then be pronounced for the child. Once the child successfully pronounces the Hindi word, an image associated with the word is displayed in front of the child to make appropriate retentions in mind. The user interface will be easy to operate by the teacher so that one can update the required list of words easily. This system will be one of its kinds, pioneering the way in how to teach dyslexic children. Here, the system would use a stored dictionary of two- or three-letter Hindi words. The main GUI, which can be used by the teacher or therapist, will support the playing of a recorded Hindi audio. All the audio words will be from the stored dictionary. Each word will be pronounced to the child. Then, the system will listen to the child repeating the word back. If the child is able to repeat the word with correct pronunciation, the system would automatically move to the next random Hindi word. However, in case of failure, the system will repeat the word slowly unless the child is able to repeat it. Eventually, the system will help the child make a better retention of the word in his mind by displaying an appropriate image for the word. All the images for the words will also be stored in a database locally so that the process of pulling up an image is faster. The system will work on top of an easy-to-use GUI built on Python Tkinter [17].

3.1 Traditional generative adversarial network

The most basic architecture of a GAN includes a generator and a discriminator. The task of the generator is to generate data from a random noise vector. The generated data are sent to the discriminator, where it is compared with the actual data, and deemed as fake or real. Once the generated data are seen as real by the discriminator, the generator starts generating similar data with variations. One of the neural net, known as the generator, creates new data nodes, while the other, the discriminator, compares them if they are real or not, i.e. the discriminator decides whether each made instance of data that it looks at is in the actual training dataset or not. In a nutshell, the main goal of the generator is to generate data [18, 19], whereas the goal of the discriminator is to identify data coming from the generator as fake. The generated data are sent into the discriminator alongside a set of data taken from the actual dataset. Then, the discriminator uses both real and fake data and provides some probabilities, a number between 0

and 1, with 1 meaning the prediction real and 0 meaning a fake. So essentially a double feedback loop is made. The discriminator is present in a feedback loop with the ground-truth of the dataset, which is known, and the generator is in a feedback loop with our discriminator. $D(x)$ represents the probability that x came from the original data rather than the generators distribution p_g , so D is trained to maximize the probability of assigning the correct label to both training examples and samples from G . G is also simultaneously trained to minimize In simpler terms, D and G play the following two-player minimax game with value function $V(G, D)$ as shown in (1):

$$\min(\max V(D, G) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{1}$$

3.2 Workflow

As in Figure 1, if the child is unable to pronounce the Hindi word correctly, the system would repeat the word for him. If the child is still unable to get the word correctly, the system would break the Hindi word into syllables. Syllables would be displayed on the screen, and then, using a text to speech engine, they would pronounce for the child. The child has to then pronounce the word syllable-wise

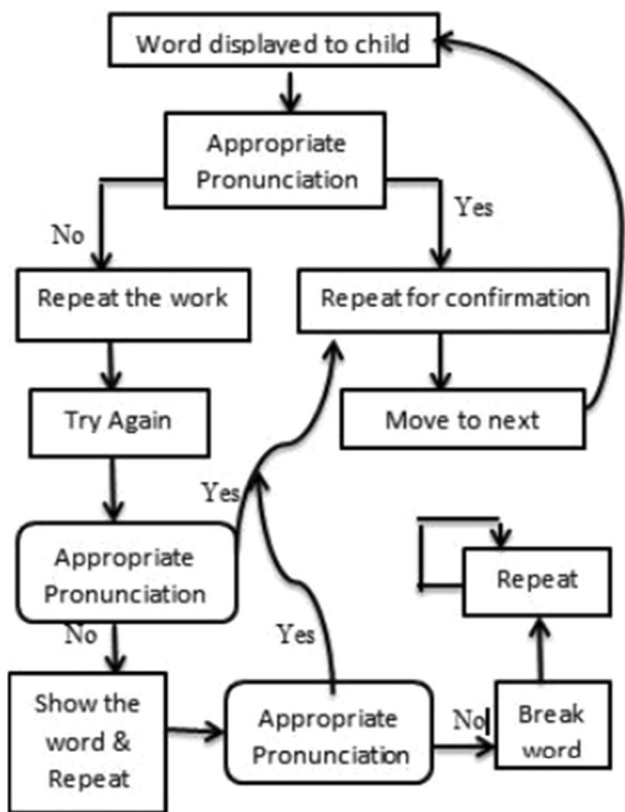


Fig. 1 Proposed architecture

syllable before he can move ahead to the next word. This process of repeated pronunciation will help the child make better retentions. To help the same cause, an image will also be shown for the particular word.

3.3 MelGAN architecture

3.3.1 MelGAN generator architecture

The generator layer used consists of a convolution layer, followed by 2 up-sampling layers. This is done to accommodate the $256 \times$ lower temporal resolution of the Mel spectrogram. Each of the convolution layers is then stacked by residual blocks which have dilated convolutions. Unlike vanilla GANs, generator in this architecture does not use a random global noise vector as an input; instead, it uses Mel-spectrogram. The generator is designed such that the inductive bias is put so that there is long range correlation between the audio time steps. In CNN-based generators, there exists an inductive bias which results in making the pixels which are close by spatially correlated. Each layer of up-sampling is a transposed convolution layer and has kernels which are of size twice of the actual stride. Also, each residual dilated convolution stack consists of 3 layers with a dilation of 1, 3 and 9 and a kernel-size of 3, making a total receptive field of 27 time-steps. Leaky-ReLU [20] is used as an activation function. The generator architecture is displayed in Fig. 2

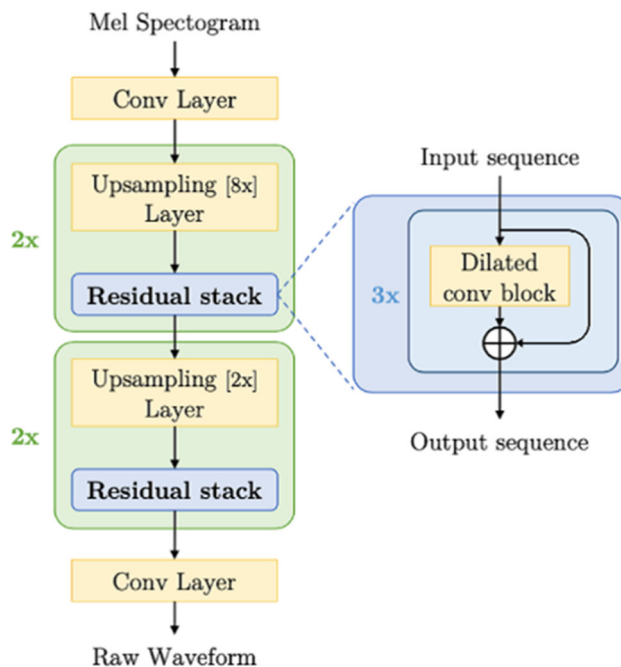


Fig. 2 Mel-GAN generator architecture

3.3.2 MelGAN discriminator architecture

For MelGAN, a multi-scale architecture with 3 discriminators (D1, D2, D3), with similar network structure but operate on different audio scales, was used. Our first discriminator D1 operates on the scale of raw audio, whereas the other two D2, D3 operate on raw sample audio which is downsampled by a factor of 2 and 4, respectively. This process is performed by using a strided average pooling with kernel size of 4. This kind of an approach allows the structure to have an inductive bias, i.e. the discriminator working on downsampled audio, does not have the rights to access any high-frequency component of the sample audio. Contradictory to a standard GAN discriminator, this one tries to learn to classify between distributions of entire audio series, whereas the window-based discriminator learns to classify between distributions of small chunk of audio. Similar to the generator, weight normalization is used in all layers of the discriminator. The first discriminator D1 gets the raw audio in waveform. After average pooling, the audio is sent to the second and third discriminator. Each discriminator consists of a convolution layer, four downsampling layer and then another convolution layer for feature mapping [21]. Each layer uses LeakyReLU for activation. The discriminator architecture is presented in Fig. 3.

3.4 WaveGAN and DCGAN

DCGAN is one of the most used architecture for generating images using a dataset of sample images. Using it as basis for WaveGAN is only a natural approach. Generally, DCGAN generator uses the transposed convolution

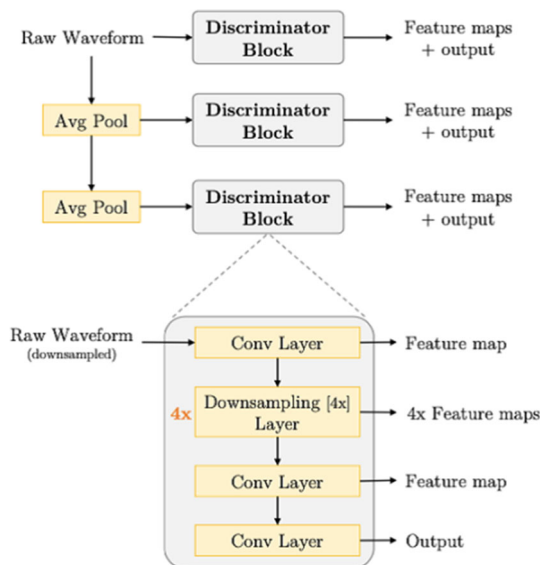


Fig. 3 MelGAN discriminator architecture

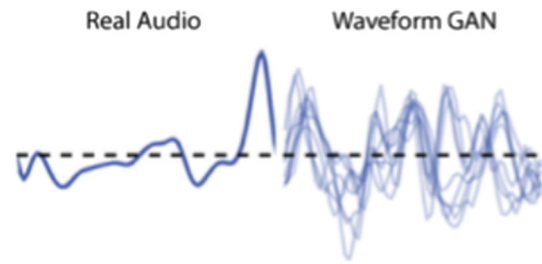


Fig. 4 Wave-GAN resulting waveform

operation [21], as shown in Fig. 4, to iteratively up-sample a set of feature maps with low resolution into images with high resolution. Therefore, the transposed convolution operation is modified to widen the receptive field. So, instead of using a two-dimensional filter of 5×5 , a one-dimensional filter of length 25 is used. Similarly, the transposed convolution on 5×5 2D filter discriminator is also modified such that one-dimensional filters of length 25 are used instead of two-dimensional filters of size 5×5 , and up-sampling is done by a factor of 4 instead of 2 at each layer. These modifications lead to WaveGAN in having the same number of operations numerically and then output the same dimensionality as from a DCGAN. Since DCGAN outputs images with 64×64 pixels—which is equivalent to just 4096 audio samples—an added layer to the architecture is there, resulting in 16,384 samples, slightly more than a single second of audio at 16 kHz frequency.

3.5 Dataset specification

The dataset used for testing WaveGAN is SC09, or Speech Commands Zero through Nine. The dataset used is a subset of the actual dataset built by Google in 2017. The actual dataset contains voice commands which will help one train voice interfaces for applications and devices. The Google Speech Commands Dataset was made by the Tensor Flow and AIY teams to showcase the speech recognition examples using the new Tensor Flow API. The dataset consists of 65,000 clips of one-second long duration. Each clip contains one of the 30 different words spoken by thousands of different people. The clips were recorded in realistic environments with the help of phones and laptops. There are 35 words that contained noise words, and also the ten command words most useful in a robotics environment are present [22].

3.6 System specification for implementing WaveGAN

WaveGAN requires a lot of resources to run properly. It is primarily run on a GPU with at least 4 GB video RAM.

Using the SC09 dataset, WaveGAN was ran on Google Colab Notebook Platform. The whole set-ups had the following specification [23].

3.7 MelGAN implementation

MelGAN is the second approach that this studies follow in generating audio from raw audio samples. This approach allows to generate audio samples which are longer than 1 s. MelGAN is feedforward convolution architecture which can perform audio waveform generation. It is also capable of yielding high-quality TTS synthesis results.

3.8 Comparing audio voice input and audio samples

A two-step comparison process will take place once the system has recorded child's audio. The following steps will take place.

1. Using Google TTS Engine to convert the recorded audio to text and then compare the audio sample text and the converted text. This is a root-level approach that would ensure faster working.
2. The second approach finds the relative degree of similarity between the audio profiles. Using audio profiling, fingerprints can be generated. An audio fingerprint is like a condensed digital summary for any audio file, which can be used to identify the audio sample.

4 Result and discussion

4.1 WaveGAN result

WaveGAN gave some very interesting results when applied on SC09. The program that ran for the whole cut-off time period was only able to provide partial results. Moreover, the generated audio samples were very low in fidelity as only 500 epochs were successfully running before the Colab environment cut off. Figure 4 shows the waveform output of the actual audio and generated audio. As shown, if enough epochs, say 2000, were possible, the generator was gradually moving towards the actual waveform of the audio sample. The program took 12 h to run.

Comparing the frequency distribution, as shown in Fig. 5, of the generated audio samples shows that the fidelity of generated samples was very low. Some factors which affect fidelity are frequency response, distortion, noise and time-based errors. It could be attributed to low epoch count as well as noise present in the raw sample audio as well. Moreover, the generated audio samples

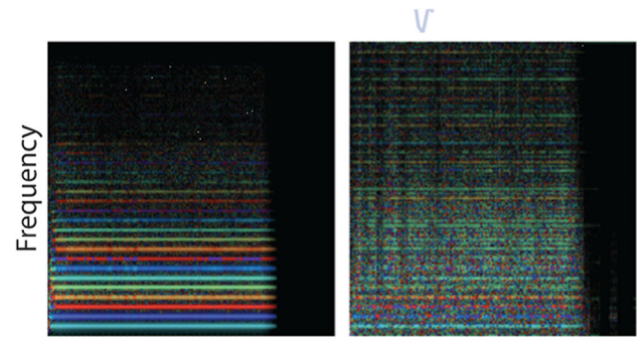


Fig. 5 Frequency comparison between real audio (left) and generated audio (right)

worked well only for 1-s slices of audio [24]. These 1-s slices were only able to capture basic two letter words which consisted of only single syllables. Our purpose of generating audio samples for two- or three-letter Hindi words was not met by WaveGAN. To overcome the 1-s audio time limit, MelGAN was used. Figure 5 shows frequency comparison. As figure gives real and generated frequency of audio input for two- and three-letter Hindi words, it shows that WaveGAN is generating audio for three-letter words and it does not gives quality audio output for more than 1 s of audio input. Its output quality gets degraded as size of word get increased. But while generating audio for two-letter words which is of 1 s, it gives best output. From previous examples, WaveGAN achieves more inception score around 6.0 [25]. WaveGAN is labelled more accurately by human around 66%. However, while considering criteria of sound quality and speaker diversity, humans indicate a preference for WaveGAN. For drum sound effects, WaveGAN captures semantic modes such as kick and snare drums. On bird vocalizations, WaveGAN generates a variety of different bird sounds. On piano, WaveGAN also generates musically consonant motifs that, as with the training data, represent a different key signatures and rhythmic patterns. A large-vocabulary speech dataset like TIMIT, consisting of multiple speakers, WaveGAN, produces speech-like babbling which is somewhat the same to results from unconditional autoregressive models [7, 26]

Experiments are performed on some of the following examples by considering their various characteristic to generate sound waveforms.

1. Drum sound effects: Drum samples for kicks, snares, toms, and cymbals for 0.7 h.
2. Bird vocalizations: In-the-wild recordings of many species for 12.2 h [27]
3. Piano: Professional performer playing a variety of Bach compositions for 0.3 h.
4. Large speech dataset (TIMIT) for 2.4 h: Multiple speakers [22]

WaveGAN networks converge by their early stopping inception score or stopping criteria in four days (around 200 k iterations, around 3500 epochs), and produce speech-like audio within the first hour of training. Above example represents accuracy of WaveGAN.

4.2 MelGAN result

MelGAN achieved promising results for our dataset. To counter our resource problems, MelGAN was ran on a Google Cloud virtual machine instance, whose configuration is in Table 1. While making comparison between two waveforms, degree of similarity is measured and that degree of similarity is set between 0 and 100. If it is zero, it means both waveforms are completely different. If the degree of similarity is 100, it means waveforms are completely similar.

While calculating similarity between two waveforms, some steps need to be followed: first id preprocessing step, in which outliers are removed from trend data then smoothing and compressing steps are performed. In outlier removal step, when any of the value deviates too much from previous value or any other values, then that value will be set as equal to the previous one. In smoothing step, weighted average is calculated. In compression step, an average of some data points is taken. Then, both waveforms are sorted in such a way that similarity can be calculated. Here, on comparing the actual waveform with the waveform of the generated audio, the similarity score was found to be 89% which is more as compared to WaveGAN. Both the waveforms, original and the generated, can be seen in Figs. 6 and 7, respectively. The main difference is the ability of the GAN architecture to duplicate the background sound which also leads to the dissimilarity.

4.3 Evaluation method

To evaluate how MelGAN performed compared to other model previously used, samples were generated from each model. Table 4 shows the comparison of various models used. As evident, MelGAN performs exceptionally well. Evaluation metric mean opinion score (MOS) was also used to compare the performance of the model to other architectures.

Table 2 compares MOS for different architectures. The samples were taken at random and then scored from 1 to 5. The samples were given and rated one at a time by the

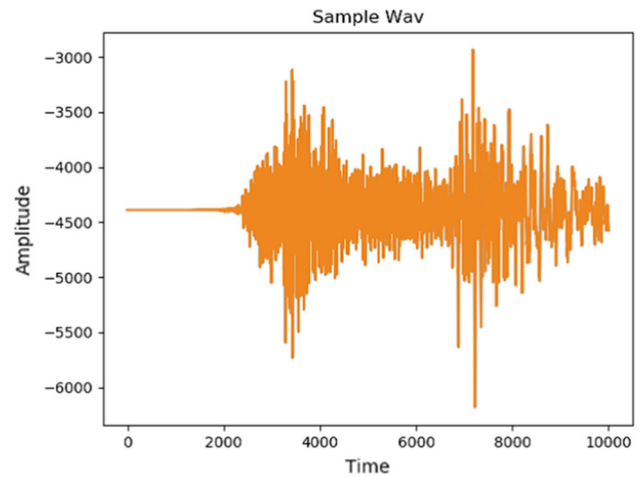


Fig. 6 Original waveform

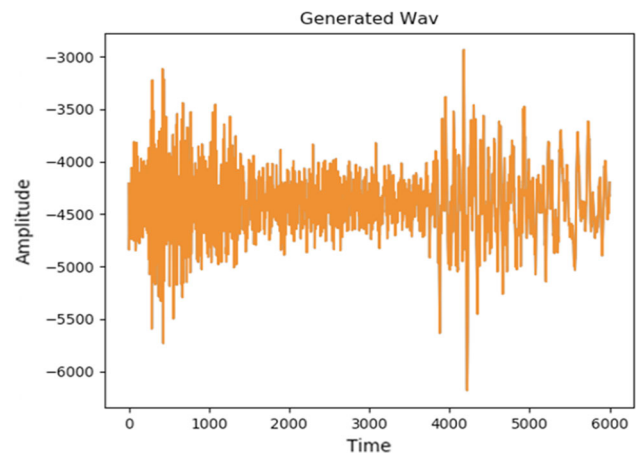


Fig. 7 Generated waveform

Table 2 Comparison by using MOS

| Model | MOS |
|-----------|------|
| Wave glow | 3.52 |
| WaveGAN | 3.72 |
| MelGAN | 4.11 |
| Original | 4.46 |

testers. After collecting all the evaluation results, the MOS score μ_i of model i is estimated by averaging the scores m_i , of the samples coming from the other models as shown in (2)

Table 1 System specification

| CPU | RAM | SPACE | Cut-of-Time | GPU |
|-----------------|-------|-------|-------------|---------------|
| 2vCPU @ 2.2 GHz | 13 GB | 64 GB | 12 h | 1 × Tesla K80 |

Table 3 Cloud virtual machine configuration

| Virtual machine configuration | |
|-------------------------------|----------------------------|
| CPU | 8v CPU standard-el |
| RAM | 30 GB |
| GPU | NVIDIA Tesla T4 12 GB VRAM |
| Storage space | 200 GB |

$$\mu_i = \frac{1}{N_i} \sum_{k=1}^{N_i} m_{i,k} \quad (2)$$

The MOS closer to the original score or 5, whichever is more, means the generated audio has been able to successfully fool the discriminator and is the nearest copy of the actual samples.

4.4 Generated audio samples

When MelGAN was run on top of Google Cloud Platform, it was able to produce a set of audio sample in about 2000 epochs which took around 4 h. Table 3 shows the specifications of the virtual machine instance used. Table 4 shows the comparison of other techniques with MelGAN with respect to the number of epochs and time taken.

In Table 4, comparison of audio synthesis techniques is given. As shown in Table 4, MelGAN outperforms the other models at the moment. It was able to provide the best fidelity rate amongst all and went up to 2500 epochs in just 8 h. The samples generated by MelGAN at an epoch jump of 500 can be seen in Fig. 8. After about 500 epochs, each of the sample gets modified closer to the real audio sample. Since MelGAN does not use random noise vectors at the beginning of the generator process, it is quite quicker compared to other models. Also the generator is able to learn the waveform quicker due to it being fed as inversed Mel-spectrograms.

Table 4 Comparison of audio synthesis techniques

| GAN | CPU | GPU | Time (h) | Epochs | Platform |
|---------|-----|-----|----------|--------|----------|
| WaveGAN | 2 | – | 20 | 500 | MacOS |
| WaveGAN | 4 | 1 | 12 | 500 | Colab |
| WaveNet | 4 | 1 | 12 | 800 | Colab |
| MelGAN | 8 | 1 | 8 | 2000 | GCP |

5 Conclusion and future scope

Generative adversarial networks have majorly been researched in terms of image generation. Many GANs over the years iteratively improved the original GAN presented by Ian Goodfellow. Eventually, researches diversified their approach towards GAN to other data types such as audios and videos. Using WaveGAN, one of the first auto-regressive models purely made to generate synthetic audio provided some acute insights into the process. However, due to its low fidelity, WaveGAN was not a good fit for the system. It is very important for such a system to provide high-fidelity output, so that the child listening to the system can learn the correct pronunciations. WaveGAN also failed at generating audio sample greater than 1 s and only made noisy samples for input audio greater than 1 s. Google's Deep Mind project brought forward another possible solution for audio generation. They developed WaveNet project which used GAN architecture at a scalable level to provide a text to speech engine. This engine acted as the base for getting our audio sampled since it supported Hindi audio. WaveNet audio generation system is part of the cloud services that Google offers. Hence, the system makes use of the TTS engine on the go. However, for generating audio from random noise vector, WaveNet requires a lot of computing resource which could only be provided by a cloud virtual machine instance. Hence, WaveNet was only used a root level for TTS task. Over the coming years, researchers started looking into audio sampling for music compositions and human speech synthesis more clearly. Various ways which included parallel data or non-parallel audio translation for speech synthesis came forward. One of the major developments was implementation of a model called MelGAN. MelGAN, unlike other architectures for audio synthesis, used inversed Mel spectrograms for generation, whereas WaveGAN and SpecGAN used waveforms MelGAN focused on modified DCGAN approach with spectrograms to synthesize audio. One of the major advantages of MelGAN was that it is not resource-intensive. Even on a CPU, MelGAN can produce audio with comparable high fidelity in an hour. Such a model was perfect for our system, since the audios generated need to be done quickly. MelGAN also provided high fidelity compared to WaveGAN with MOS. It was also able to generate audio samples greater than 1 s, hence fitting all our requirements for audio generation. Instead of using instance normalization, like most of the GAN models, weight normalization was chosen for MelGAN. Such an approach helped the model increase the quality of the generated samples. The GUI also needs to be simple enough to be understood and used by an instructor. Moreover, it should not overwhelm the child with more

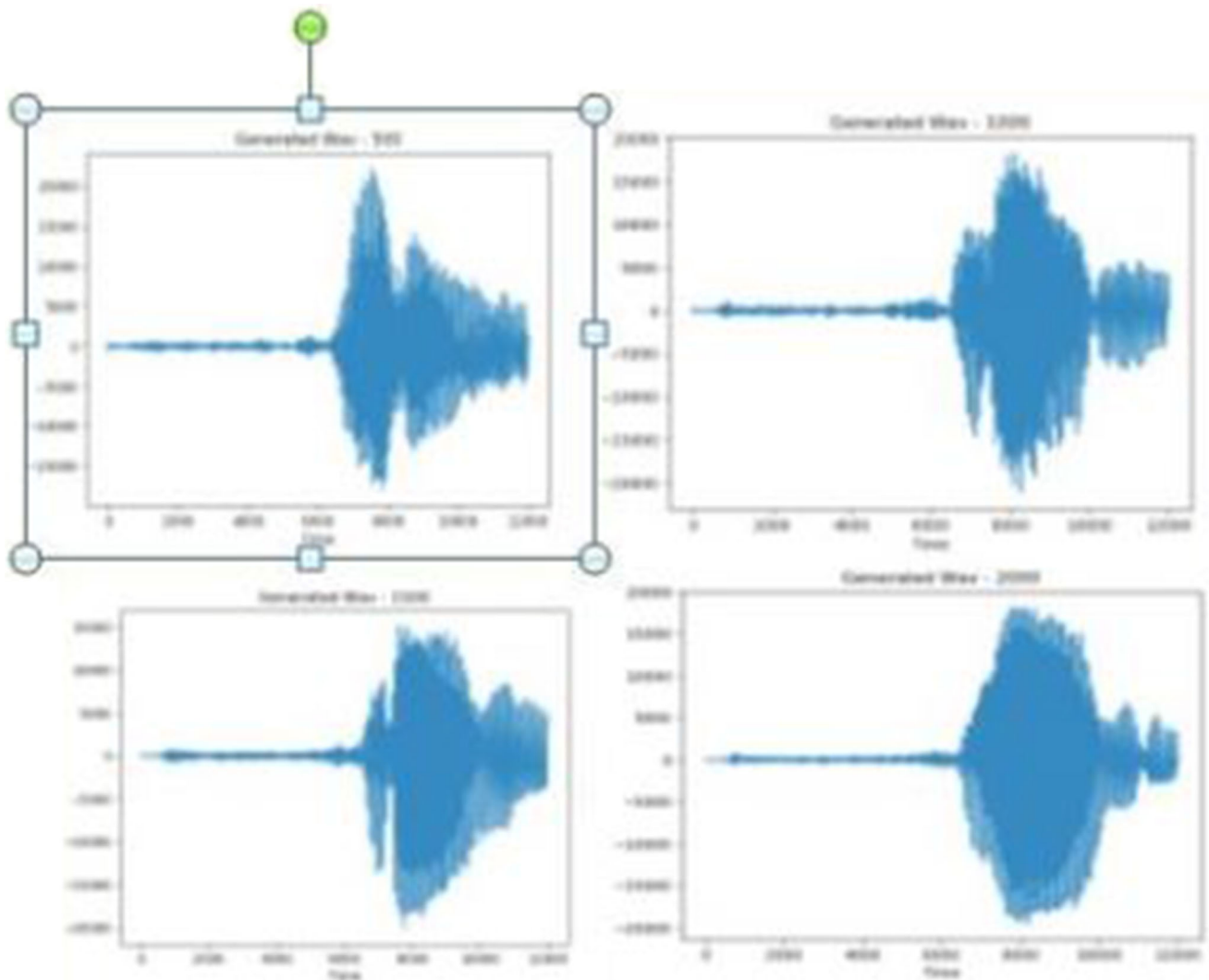


Fig. 8 Generated audio waveform at 500, 1000, 1500, 2000 epoch

information than necessary, so the GUI is approached with a basic clean look, so that the child can easily look at the word at hand, listen the audio clearly and react accordingly. Audio output using PyGame [28] library helped the system to integrate Tkinter GUI set-up and PyGame audio library seamlessly [29]. During several steps in training, the number of filter and layers was adjusted to achieve a satisfactory output. The changes made help improve the audio generation time but decreased the fidelity of audio. To counter that, an attempt at demonising the generated audio was made but that made the word impossible to hear. Since it moved further from our purpose of generating clear audio samples, the number of filters an layer used was restored to original. In the current era of Covid-19, research is improved and focussed [30, 31]

Compliance with ethical standards

Conflict of interest “Maharashtra Dyslexia Association”, government organization for dyslexic children, helped to get knowledge about dyslexic children, and their actual experience helped to develop system.

References

1. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680
2. Hecht-Nielsen R (1992) Theory of the back propagation neural network. In: Neural networks for perception. Academic Press, pp 65–93
3. Donahue C, McAuley J, Puckette M (2018) Adversarial audio synthesis. Preprint <http://arxiv.org/abs/1802.04208>

4. Isola P, Zhu J-Y, Zhou T, Efros A (2017) A Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1125–1134
5. Oord AVD, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kavukcuoglu K (2016) Wavenet: a generative model for raw audio. Preprint <http://arxiv.org/abs/1609.03499>
6. Zhang Z (2018) Improved Adam optimizer for deep neural networks. In: IEEE/ACM 26th international symposium on quality of service (IWQoS), IEEE, pp 1–2
7. van den Oord A, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior A, Kavukcuoglu K (2016) WaveNet: a generative model for raw audio. <http://arxiv.org/abs/1609.03499>
8. Zhao Y, Takaki S, Luong HT, Yamagishi J, Saito D, Minematsu N (2018) Wasserstein GAN and waveform loss-based acoustic model training for multi-speaker text-to-speech synthesis systems using a WaveNet vocoder. *IEEE Access* 6:60478–60488
9. Fang F, Yamagishi J, Echizen I, Lorenzo-Trueba J (2018) April. High-quality nonparallel voice conversion based on cycle-consistent adversarial network. In: 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, pp 5279–5283
10. Kaneko T, Kameoka H (2018) CycleGAN-vc: non-parallel voice conversion using cycle-consistent adversarial networks. In: 2018 26th European signal processing conference (EUSIPCO), IEEE, pp 2100–2104
11. Bińkowski M, Donahue J, Dieleman S, Clark A, Elsen E, Casagrande N, Simonyan K (2019) High fidelity speech synthesis with adversarial networks. Preprint <http://arxiv.org/abs/1909.11646>
12. Viswanathan M, Viswanathan M (2005) Measuring speech quality for text-to-speech systems: development and assessment of a modified mean opinion score (MOS) scale. *Comput Speech Lang* 19(01):55–83
13. Kumar K, Kumar R, de Boissiere T, Geste L, Teoh WZ, Sotelo J, Courville AC (2019) MelGAN: generative adversarial networks for conditional waveform synthesis. In: Advances in neural information processing systems, pp 14881–14892
14. Yoshimura T (2002) Simultaneous modeling of phonetic and prosodic parameters, and characteristic conversion for hmm-based text-to-speech systems.
15. Moulines E, Charpentier F (1990) Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech communication*
16. Chan C, Ginosar S, Zhou T, Efros A (2018) Everybody dance now. Preprint <http://arxiv.org/abs/1808.07371>
17. Lundh F (1999) An introduction to tkinter. [www. Pythonware.-com/library/tkinter/introduction/index.html](http://www.pythonware.com/library/tkinter/introduction/index.html)
18. Kaneko T, Takaki S, Kameoka H, Yamagishi J (2017) Generative adversarial network-based postfilter for STFT spectrograms. In: INTERSPEECH, pp 3389–3393
19. Radford A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks. Preprint <http://arxiv.org/abs/1511.06434>
20. Zhang X, Zou Y, Shi W (2017) Dilated convolution neural network with LeakyReLU for environmental sound classification. In: 2017 22nd international conference on digital signal processing (DSP), IEEE, pp 1–5
21. Namatėvs I (2017) Deep convolutional neural networks: Structure, feature extraction and training. *Inf Technol Manag Sci* 20(1):40–47
22. Garofolo JS, Lamel LF, Fisher WM, Fiscus JG, Pallett DS, Dahlgren NL, Zue V (1993) TIMIT acoustic-phonetic continuous speech corpus. Linguistic data consortium.
23. Cvoki D (2020) Cutting testing costs by the pooling design. *Vojnotehnicki glasnik/Military Technical Courier* 68(4):743–759. <https://doi.org/10.5937/vojtehg68-28078>
24. Kalchbrenner N, Elsen E, Simonyan K, Noury S, Casagrande N, Lockhart E, Kavukcuoglu K (2018) Efficient neural audio synthesis. Preprint <http://arxiv.org/abs/1802.08435>
25. Stevens SS, Volkman J, Newman EB (1937) A scale for the measurement of the psychological magnitude pitch. *J Acoust Soc Am* 185:83–90
26. Theis L, van den Oord A, Bethge M (2016) A note on the evaluation of generative models. In: ICLR
27. Boesman P (2018) https://www.xeno-canto.org/contributor/OOE_CIWCSWV, 2018. Accessed
28. Porter A (2013) Evaluating musical fingerprinting systems, Doctoral dissertation, McGill University Libraries
29. McGugan W (2007) Beginning game development with Python and Pygame: from novice to professional. Apress
30. Fabiano N, Radanovich S (2020) On covid-19 diffusion in Italy: data analysis and possible. *Vojnotehnicki glasnik/Military Technical Courier* 68(2):216–224. <https://doi.org/10.5937/vojtehg68-25948>
31. Salimans T, Kingma DP (2016) Weight normalization: a simple reparameterization to accelerate training of deep neural networks. In: Advances in neural information processing systems, pp 901–909

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.