IMPORTANTAUG: A DATA AUGMENTATION AGENT FOR SPEECH

Viet Anh Trinh[1]*, Hassan Salami Kavaki* [1] *and Michael I Mandel*[1,2]

[1] The Graduate Center, CUNY, New York, USA
[2] Brooklyn College, CUNY, New York, USA
vtrinh@gradcenter.cuny.edu, hsalami@gradcenter.cuny.edu, mim@sci.brooklyn.cuny.edu

ABSTRACT

We introduce ImportantAug, a technique to augment training data for speech classification and recognition models by adding noise to unimportant regions of the speech and not to important regions. Importance is predicted for each utterance by a data augmentation agent that is trained to maximize the amount of noise it adds while minimizing its impact on recognition performance. The effectiveness of our method is illustrated on version two of the Google Speech Commands (GSC) dataset. On the standard GSC test set, it achieves a 23.3% relative error rate reduction compared to conventional noise augmentation which applies noise to speech without regard to where it might be most effective. It also provides a 25.4% error rate reduction compared to a baseline without data augmentation. Additionally, the proposed ImportantAug outperforms the conventional noise augmentation and the baseline on two test sets with additional noise added.

***Index Terms*—** Data augmentation, importance maps, speech recognition, noise robustness.

1. INTRODUCTION

Data augmentation techniques are used to enhance models' performance by adding additional variations to the training data. These techniques are widely applied to improve automatic speech recognition (ASR) performance [1–4]. In [1], the authors used speed perturbation to create new speech utterances by changing the frequency components and number of time frames of speech recordings. This additional training data helped to decrease the word error rate (WER) by 3.2% relative on Librispeech task with 960 hours Librispeech data. In [2], reverberation was added to the speech to make it more realistic. Recently, a common technique is to remove or mask information in the spectrogram domain. For instance, SpecAugment [5] removes speech information in T continuous random time frames or F frequency bins. At the time, this augmentation not only increased ASR accuracy, but also achieved the state-of-the-art WER on the LibriSpeech 960-hour dataset at 5.8%. [3] proposed data augmentation via adding additional noise to speech, reducing WER by 21.3% relative on their self-constructed 100 sentence evaluation set.

Recently, data augmentation techniques have been introduced that utilize importance or saliency maps. There are many methods to predict importance and saliency maps, e.g., [6–16], but few previous studies have investigated applications of such maps. In the visual domain, a recent work [17] used saliency maps for data augmentation. Instead of using noise, the authors cut random rectangles out of an image if the sum of the importance scores of all the pixels inside the rectangle was smaller than a threshold. In speech, [18] used a bottom-up approach to predicting auditory saliency maps to improve ASR performance. They used Gabor filters to extract intensity and contrast in time and frequency to find the saliency maps. This saliency map is then multiplied with the spectrogram, resulting in a weighted spectrogram, from which features are extracted for ASR. This approach achieved a 5.3% relative WER reduction compared to a baseline that did not use importance maps.

We introduced a top-down adversarial approach to predicting importance maps in [15, 19]. The current paper builds upon those approaches to introduce a method of using our top-down importance maps for data augmentation in speech command recognition. In contrast to [18], we use a top-down approach to identify the regions that are important for recognizing the specific production of the specific words in a given utterance. Furthermore, these regions are directly related to the speech recognition task, which is different from bottom-up approaches, which produce the same prediction regardless of the task. For instance, a bottom up approach using intensity filters might predict that a spectrogram area containing loud noise is important for the speech recognition task.

In section 2, we discuss our ImportantAug[1] method, where we first identify the importance maps and then utilize them to augment the data. In section 3, we present our experimental setup with details about the data, hyperparameter settings, and experiments. The results on clean, in domain noisy, and out-of-domain noisy test sets are illustrated in section 4.

2. METHOD

The proposed network has a speech command recognizer and a mask generator, as illustrated in Figure 1. The speech command recognizer's task is to classify the input utterances into the correct classes. The mask generator's task is to add as much noise as possible to utterances without harming the performance of the recognizer. This has the effect of generating importance maps, which are utilized for data augmentation.

Our networks are trained in two stages. In the first stage, we train the generator so that it can output importance maps (masks). We load a recognizer that is pre-trained on clean speech. Then, we freeze the recognizer and train only the mask generator. The generator receives clean speech as input and outputs a mask. This mask is multiplied with the noise and then added to the clean speech, resulting in a noisy utterance. The recognizer receives this noisy speech as input and predicts a class. Note that in the Google Speech Commands (GSC) dataset [20], each utterance is at most 1s long and only contains a single word in the presence of noise. Thus this is a speech classification task as opposed to a full speech recognition task.

We designed the loss function for our network to encourage the mask to maximize the amount of noise while the speech recognizer maintains good performance. This loss function therefore forces

[1]The code is available at https://github.com/tvanh512/importantAug

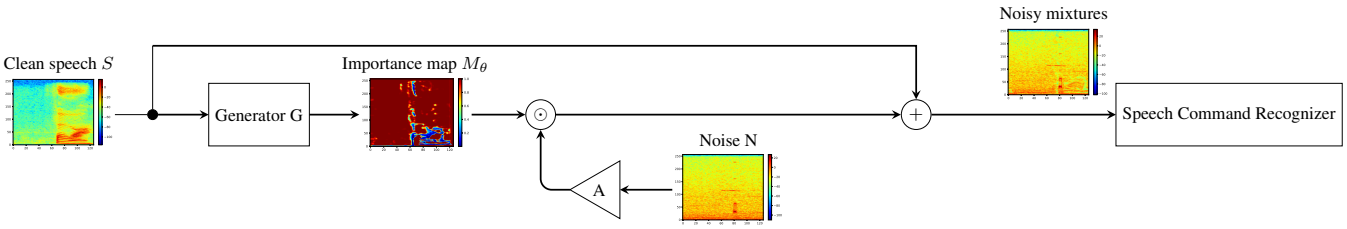


Fig. 1. ImportantAug scheme. The mask generator's task is to output an importance map (mask) for an utterance with maximal noise while interfering with recognition of the recognizer as little as possible. The mask is point-wise multiplied ($\odot$) with the scaled noise and added to the clean speech. The mask contains values close to 0 at important points and values close to 1 at unimportant points.

the generator to output a mask with less noise in regions that are important to the recognizer, and with more noise in regions that are unimportant to the recognizer.

In the second stage, we freeze the generator and train only the speech command recognizer. We aim to create additional data to train the recognizer. To create additional data, noise is added to the unimportant regions of the clean speech. Less or no noise is added to the important regions.

Denote $S(f,t)$ and $N(f,t)$ as the complex spectrograms of the speech and noise, respectively, where f is the frequency index and t is the time index. These spectrograms are created by applying the short time Fourier transform (STFT) to the time domain signal $s(t)$ of the speech and $n(t)$ of the noise. The generator G with parameters θ takes $\tilde{S}(f,t) = 20\log_{10}|S(f,t)|$ as input and predicts a mask $M_\theta(f,t)$ with the same shape as $\tilde{S}(f,t)$

$$M_\theta(f,t) = G(\tilde{S}(f,t);\theta) \in [0,1]^{F\times T} \tag{1}$$

An additional augmentation shifts the mask slightly in time or frequency to further increase variability in the training data for the recognizer. The mask output by the generator, M_θ, is rolled along the frequency and time dimension

$$M_{\theta r} = r(M_\theta;\delta) \tag{2}$$

where r is the roll operator (we use torch.roll) and δ is the number of time frames or frequency bins by which the elements of the mask are shifted. δ is drawn uniformly at random from the interval $(-D, D)$. Furthermore, to create additional variation, with probability 0.5, the mask $M_{\theta r}$ is replaced by a mask of all 1's. Denote whichever mask is selected as M. This rolling augmentation is only used when re-training the recognizer using the predicted importance maps and not when training the mask generator itself.

This mask is then applied point-wise to a noise instance N, scaled by gain A. The gain A is adjusted each training batch such that the signal to noise ratio is maintain at a target value

$$A = \sqrt{\frac{\sum_{b,t,f}|S_{btf}|^2}{10^{v/10}\sum_{b,t,f}|N_{btf}|^2}}, \tag{3}$$

where v is the target SNR expressed in decibels, and b, t, f denoted the batch, time, and frequency dimensions respectively. The resulting masked-scaled noise $AN \odot M$ (where $\odot$ denotes point-wise multiplication) is added to the clean speech S. The resulting noisy mixture is input to the speech command recognizer R, which predicts the probability of the class $\hat{y}$

$$\hat{y} = R(S + AN \odot M). \tag{4}$$

The model is trained to minimize

$$\mathcal{L}(\theta) = \lambda_r \mathcal{L}_{\mathrm{R}}(y,\hat{y}) - \frac{\lambda_e}{TF}\sum_{f,t}\log M + \frac{\lambda_f}{TF}\sum_{f,t}|\Delta_f M| + \frac{\lambda_t}{TF}\sum_{f,t}|\Delta_t M|. \tag{5}$$

where $\mathcal{L}_{\mathrm{R}}$ is the loss of the speech recognizer, Δ_f is the difference operation along frequency, Δ_t is the difference operation along time, and $\lambda_r, \lambda_e, \lambda_f$, and λ_t are weights set as hyperparameters of the model. The recognizer loss is the cross entropy between the prediction $\hat{y}$ and the ground truth label y. This loss forces the recognizer to keep high accuracy on predicting the correct class. The $-\sum_{f,t}\log M$ term forces the mask's value to be close to one, thus maximizing the amount of noise added. The terms associated with λ_f and λ_t encourage the mask to smooth in frequency and time.

3. EXPERIMENTAL SETUP

3.1. Dataset

We use the Google Speech Commands (GSC) dataset version 2 [20] for our experiments. This dataset includes 105,829 single-word utterances of 35 unique words. Many utterances include noise or other distortions. The models were trained on the training set and evaluated on the test set. The development set was used for early stopping.

We also employ additional noise from the MUSAN dataset [21] to augment the speech from the GSC dataset. The recordings in MUSAN have different lengths, so we only used the first second from each recording and exclude any recordings shorter than one second as the speech utterances are restricted to be at most one second long. There are 877 noisy files after filtering out short utterances. We randomly choose 702 files (80%) for training. We mix the remaining 175 files with the utterances from the GSC test set, creating a new noisy test set that we call GSC-MUSAN.

To evaluate our trained model on out-of-domain noisy environments, we also create another test set. First, we select a file "HOME-LIVINGB-1.wav", which contains 40 minutes of noise recording in the living room environment from the QUT corpus [22]. We then resample this file from 48 to 16 kHz, the same rate as the GSC utterances and choose random sections in this noisy file to mix with the utterances in the GSC test set. We call this dataset GSC-QUT.

3.2. Experiments

We compare our proposed method against two other methods. In the first method (baseline), we train a recognizer that does not utilize any data augmentation . It is trained on the GSC training set and selected

Table 1. Recognizer error rate (%) on the Google Speech Command v2 (GSC) development set with conventional noise augmentation at different SNRs

SNR	Dev	SNR	Dev
∞	7.74	15	**5.83**
40	6.39	10	6.11
35	7.65	5	6.00
30	6.10	0	5.97
25	6.19	-5	6.24
20	6.22	-10	6.16

using early stopping on the development set. All other methods are trained by initializing their parameters to those of this pre-trained baseline recognizer. In the second method, we utilize a conventional noise augmentation technique that treats all time-frequency points as equally important and applies noise directly to the speech without importance maps (S + AN). We perform an experiment to identify the best single signal to noise ratio (SNR) to use, comparing those ranging from -10 dB to 40 dB in steps of 5 dB. We also evaluate ∞ dB by training on clean data.

In our proposed method, ImportantAug, we performed the two-stage training as described above. First, we load and freeze the recognizer from the baseline and train the generator. Then, we freeze the generator and train the recognizer. The noise from the MUSAN dataset was multiplied with the rolled importance maps and added to the speech. In addition, we perform an ablation study by evaluating the recognizer performance when we remove the importance map from the proposed approach, by setting the mask to be all 1's, which we call the "Null ImportantAug" condition. In this case, no region is more important than other regions and the noise is added directly to the speech. We evaluate the baseline (no augmentation), conventional noise augmentation, ImportantAug and Null ImportantAug on the standard GSC test set, GSC-MUSAN and GSC-QUT noisy test sets.

In addition to using continuous-valued importance maps, we also experimented with binarizing the importance maps . We considered the $q\%$ of time-frequency points with the lowest value in the continuous-valued importance map as being important and did not add any noise to them. The other $100 - q\%$ of the points were considered unimportant and noise was added to them. In this experiment, the mask was not replaced by an all 1's mask at all.

3.3. Hyperparameter settings

The signal was sampled at 16 kHz with a window length of 512 and a hop length of 128 samples, leading to a spectrogram with 257 frequency bins and 126 time frames for a 1 s utterance. In all experiments, we use the same default setting for the speech command recognizer, which is a neural network with 5 layers. Each layer has a 1D depth-wise and 1D point-wise convolution [23, 24], followed by SELU activation [25]. The depth-wise convolution has a kernel size of 9×9 (281.25 Hz x 96 ms), a stride value of 1, a dilation value of 1; and its inputs and outputs are both 257 channels. The point-wise convolution consists of a kernel of size 1×1 and also has inputs and outputs for size 257.

The generator is a neural network with 4 layers, where each layer is a 2D convolutional network. The first layer takes one channel in and outputs 2 channels. The second and third layers have 2 channels in their input and output. The last layer has 2 channels of input and one of output. All the layers have a kernel size of 5×5 (156.25 Hz x 64ms), a stride value of 1, a dilation value of 1 and a padding so that the output has the same height and width as the input.

Table 2. Recognizer error rate (%) with various augmentation approaches on GSC test set

Augmentation method	Initial SNR (dB)	Error
No augmentation	∞	6.70
Conventional noise augmentation	15.0	6.52
ImportantAug	-12.5	**5.00**
Null ImportantAug	-12.5	6.12

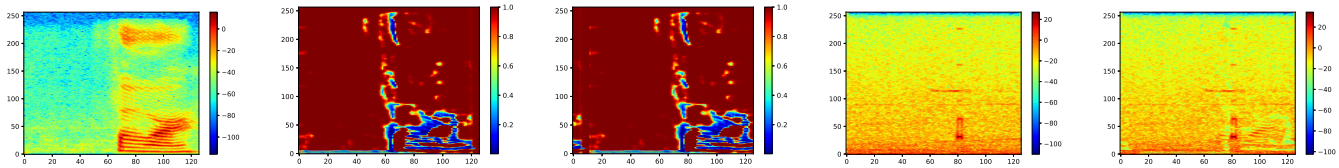
In the proposed ImportantAug method, we selected hyperparameters $\lambda_r = 1$, $\lambda_e = \lambda_f = \lambda_t = 3$, $v = -12.5$ dB. First, the weights $\lambda_r, \lambda_e, \lambda_f, \lambda_t$, and v were manually adjusted based on a very small number of settings so that the speech command recognizer performed well and the mask values were closer to all 1's on the development set. Then we chose D, the maximum number of time frames or frequency bins by which the elements of the mask are shifted to be 30, equivalent to 937.5 Hz and 264 ms. This was selected to keep the mask from shifting too far from the original position.

All the models are trained with the Adam optimizer with an initial learning rate of 0.001, which is decayed by half every 20 epochs and a batch size of 256. The models are trained for 200 epochs with early stopping on the development set loss with a patience value of 30.

4. RESULTS

Table 1 shows the error rate on the development set for the conventional augmentation method with different signal to noise ratios. We can see that adding too much noise leads to a high error rate, for example, SNRs -10 and -5 dB have error rates 6.16% and 6.24%, respectively on the development set. Adding too little noise is also not optimal, for instance, SNRs 40 and 35 dB have error rate 6.39% and 7.65% on the development set. Using no noise at all does not provide good performance, with an error rate 7.74%. However, adding the right amount of noise is beneficial for the recognizer as it balances variation in the training data with speech fidelity. As shown in Table 1, the best error rate (5.83%) is with an SNR of 15 dB. The model trained with SNR 15 dB has the best performance on the development set, so we choose this model to evaluate on the test set and compare with other approaches in Table 2.

Table 2 shows the results on the standard GSC test set. The baseline speech command recognizer has an error rate of 6.70%. The conventional noise augmentation method produces a model with an error rate of 6.52%. Our proposed method has the best error rate at 5.00%, which is a 25.4% relative improvement over the no augmentation baseline and 23.3% relative improvement over the conventional noise augmentation method. We also perform an ablation study with the Null ImportantAug method by using a "mask" that is all 1's, which leads to an error rate of 6.12%. Null ImportantAug is similar to the traditional NoiseAug because it does not utilize importance maps. The difference is that Null ImportantAug is trained with the same SNR as ImportantAug (-12.5 dB), while the traditional NoiseAug uses the SNR chosen based on the performance on the development set of 15 dB. The error rate with and without important maps are 5% and 6.12% respectively, thus the importance map is necessary for the observed performance gains.



(a) Clean speech (b) Importance map (IM) (c) Rolled IM (d) MUSAN noise (e) Noisy speech

Fig. 2. (a) Clean utterance from Google Speech Commands dataset. (b) Importance map (blue areas) from the generator. (c) Rolled importance map. (d) MUSAN noise. (e) Noisy speech created by multiplying the noise from (d) with the mask from (c) and adding clean speech from (a)

Table 3. Recognizer error rate (%) of augmentations on in-domain noise test set (GSC-MUSAN) as a function of test SNR.

	Test SNR						
Method	−12.5	−10	0	10	20	30	40
No aug. (baseline)	77.6	72.7	45.2	21.0	11.5	8.4	7.3
Noise aug. (SNR 15)	65.8	57.7	26.3	10.8	7.3	6.6	6.4
ImportantAug	**43.5**	**35.0**	**13.3**	**7.4**	**5.7**	**5.2**	**5.1**
Null ImportantAug	45.2	37.0	15.0	8.5	6.9	6.2	6.0

Table 4. Recognizer error rate (%) of augmentations on out-of-domain noise test set (GSC-QUT) as a function of test SNR.

	Test SNR						
Method	−12.5	−10	0	10	20	30	40
No aug. (baseline)	90.9	87.3	55.8	20.8	9.6	7.4	7.0
Noise aug. (SNR 15)	89.0	83.5	42.0	12.9	7.3	6.5	6.2
ImportantAug	**72.0**	**61.3**	**23.5**	**8.9**	**5.8**	**5.1**	**4.8**
Null ImportantAug	72.3	61.6	24.8	10.0	6.8	6.1	6.0

Table 3 shows the results on the GSC-MUSAN test set. We could observe that the proposed method ImportantAug achieve the best result in all SNR range. For example, the ImportantAug achieve 13.3% error rate at 0 dB, which is around one-third of the error rate of the baseline 45.2% and a half of the conventional augmentation method. We also observe that the error rates are going up if we remove the importance map (IM) when comparing row 3 and row 4 of Table 3. For example, at SNR 0 dB, the error rate going up from 13.3% to 15% if we remove the IM and train with only the noise.

Table 4 shows the results on the GSC-QUT test set, which is out-of-domain noise test set because the models are trained with MUSAN noise, not with QUT noise. Here, we observe the same trend when the ImportantAug outperforms the baseline, the conventional augmentation method.

Figure 2.b shows an example of an importance map of an utterance of the word "four" in the GSC dataset. The importance map includes the fundamental frequency, the harmonics, and the outer border shape of the speech. These regions are predicted to be necessary for the speech command recognizer to identify this specific utterance. Thus, keeping these regions clean and adding noise outside of them makes the data more diverse while not affecting the recognition.

Table 5 shows the error rate on the development and test set for the binary ImportantAug method with different important region ratios. In this experiment, we consider the quantile $q\%$ of the regions that have lowest mask value to be important. The best result is achieved on the development set by choosing 10% of points to be important, which provides a 11.3% relative error reduction on the test set compared to not multiplying the noise with the importance map ($q = 0$). Thus only a very small proportion of points need to be preserved in this way to enhance the data augmentation performance.

Table 5. Recognizer error rate (%) with binarized ImportantAug using different important region ratios, q on the original GSC test set.

q (%)	Dev	Test
70	5.42	5.64
50	5.49	5.92
40	5.19	5.71
20	5.17	5.15
10	**5.00**	5.43
5	5.09	4.92
1	5.12	4.94
0	6.03	6.12

5. CONCLUSION

In conclusion, we have demonstrated a data augmentation agent that improves a speech command recognizer. Our proposed ImportantAug method produced a 25.4% relative error rate reduction compared to the non-augmentation method and and 23.3% relative reduction compared to the conventional noise augmentation method. Taken together, this work shows that importance maps can be estimated accurately enough to be helpful for data augmentation, providing one of the first such demonstrations, especially for speech. In the future, we will extend this framework by replacing the speech command recognizer with a full large vocabulary continuous speech recognizer and we will deploy different methods to identify the importance map and use the map to augment the speech data, such as those based on human responses. The proposed method could also be used in computer vision tasks, such as image recognition by predicting importance maps for images.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation (NSF) under Grant IIS-1750383. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF.

7. REFERENCES

[1] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "Audio augmentation for speech recognition," in *Proc. Interspeech 2015*, 2015, pp. 3586–3589.

[2] Chanwoo Kim, Ananya Misra, Kean Chin, Thad Hughes, Arun Narayanan, Tara Sainath, and Michiel Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in google home," in *Proc. Interspeech*, 08 2017, pp. 379–383.

[3] Awni Y. Hannun, Carl Case, J. Casper, Bryan Catanzaro, G. Diamos, Erich Elsen, R. Prenger, S. Satheesh, Shubho Sengupta, A. Coates, and A. Ng, "Deep speech: Scaling up end-to-end speech recognition," *ArXiv*, vol. abs/1412.5567, 2014.

[4] Thai-Son Nguyen, Sebastian Stueker, Jan Niehues, and Alex Waibel, "Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation," in *Proc. ICASSP*. IEEE, 2020, pp. 7689–7693.

[5] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[6] Laurent Itti, Christof Koch, and Ernst Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[7] Jonathan Harel, Christof Koch, and Pietro Perona, "Graph-based visual saliency," in *Advances in Neural Information Processing Systems*, Cambridge, MA, USA, 2006, NIPS'06, p. 545–552, MIT Press.

[8] Saumya Jetley, Naila Murray, and Eleonora Vig, "End-to-end saliency mapping via probability distribution prediction," in *Proc. IEEE CVPR*, 2016, pp. 5753–5761.

[9] Matthias Kummerer, Thomas S. A. Wallis, Leon A. Gatys, and Matthias Bethge, "Understanding low- and high-level contributions to fixation prediction," in *Proc. IEEE ICCV*, Oct 2017.

[10] Xiaodi Hou and Liqing Zhang, "Saliency detection: A spectral residual approach," in *Proc. IEEE CVPR*. Ieee, 2007, pp. 1–8.

[11] Junting Pan, Cristian Canton Ferrer, Kevin McGuinness, Noel E O'Connor, Jordi Torres, Elisa Sayrol, and Xavier Giro-i Nieto, "Salgan: Visual saliency prediction with generative adversarial networks," *arXiv preprint arXiv:1701.01081*, 2017.

[12] Nam Wook Kim, Zoya Bylinskii, Michelle A Borkin, Krzysztof Z Gajos, Aude Oliva, Fredo Durand, and Hanspeter Pfister, "Bubbleview: an interface for crowdsourcing image importance maps and tracking visual attention," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 24, no. 5, pp. 1–40, 2017.

[13] Constantin Spille and Bernd T Meyer, "Listening in the dips: Comparing relevant features for speech recognition in humans and machines.," in *INTERSPEECH*, 2017, pp. 2968–2972.

[14] Viet Anh Trinh and Michael I. Mandel, "Large Scale Evaluation of Importance Maps in Automatic Speech Recognition," in *Proc. Interspeech 2020*, 2020, pp. 1166–1170.

[15] Viet Anh Trinh, Brian McFee, and Michael I Mandel, "Bubble cooperative networks for identifying important speech cues," *Proc. Interspeech*, pp. 1616–1620, 2018.

[16] Viet Anh Trinh and Michael Mandel, "Directly comparing the listening strategies of humans and machines," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 312–323, 2021.

[17] Chengyue Gong, Dilin Wang, Meng Li, Vikas Chandra, and Qiang Liu, "Keepaugment: A simple information-preserving data augmentation approach," in *Proc. IEEE CVPR*, 2021, pp. 1055–1064.

[18] Cong-Thanh Do and Yannis Stylianou, "Weighting time-frequency representation of speech using auditory saliency for automatic speech recognition.," in *INTERSPEECH*, 2018, pp. 1591–1595.

[19] Hassan Salami Kavaki and Michael I. Mandel, "Identifying important time-frequency locations in continuous speech utterances," in *Proc. Interspeech*, 2020, pp. 1639–1643.

[20] Pete Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[21] David Snyder, Guoguo Chen, and Daniel Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[22] David Dean, Ahilan Kanagasundaram, Houman Ghaemmaghami, Md Hafizur Rahman, and Sridha Sridharan, "The qut-noise-sre protocol for the evaluation of noisy speaker recognition," in *Proceedings of the 16th Annual Conference of the International Speech Communication Association, Interspeech 2015*. International Speech Communication Association, 2015, pp. 3456–3460.

[23] François Chollet, "Xception: Deep learning with depthwise separable convolutions," in *PRoc. IEEE CVPR*, 2017, pp. 1251–1258.

[24] Somshubra Majumdar and Boris Ginsburg, "Matchboxnet: 1d time-channel separable convolutional neural network architecture for speech commands recognition," in *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, Helen Meng, Bo Xu, and Thomas Fang Zheng, Eds. 2020, pp. 3356–3360, ISCA.

[25] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter, "Self-normalizing neural networks," in *Advances in neural information processing systems*, 2017, pp. 972–981.