Generally Applicable Deep Speech Inpainting Using the Example of Bandwidth Extension

Lars Thieling and Peter Jax
Institute of Communication Systems (IKS)
RWTH Aachen University, Germany
{thieling,jax}@iks.rwth-aachen.de

***Abstract*—Most of today's speech enhancement algorithms try to improve the quality or intelligibility of speech by modifying its time-frequency (TF) representation. It is often the case that individual parts of this TF plane become unusable due to severe disturbances or are even missing due to data loss. Here, we present a generally applicable speech inpainting algorithm to reconstruct the unusable or missing parts of the speech's TF representation in these cases. For the generalizability, we propose a statistically based error model that we use to train deep neural networks (DNNs). In order to minimize the complexity of this overall algorithm and still be able to achieve good results, we have trained the DNNs on the basis of mel frequency cepstral coefficients (MFCCs), which are designed based on the human auditory system. Our experimental results show that the proposed algorithm is well suited in reconstructing even very large unusable or missing TF parts. Using the example of artifical bandwidth extension (BWE), we demonstrate that our proposed way of training DNNs on random rectangular gaps or holes in the TF plane leads to a generally applicable solution for various specific problems in the speech processing domain.**

***Index Terms*—Speech inpainting, deep neural networks (DNNs), speech enhancement, machine learning, mel frequency cepstral coefficients (MFCCs)**

I. Introduction

Speech signals are often subject to local disturbances, which can be limited to both frequency and time. One common approach in speech enhancement, e.g., in noise reduction or echo suppression, is to attenuate these disturbances (see [1]). This is typically done by applying a spectral weight $G \in [0, 1]$ to the time-frequency (TF) representation, where G is close to 0 in regions where the disturbances are dominant and close to 1 otherwise. Especially when using a binary weighting, i.e. by setting G either to 0 or 1, this produces time- and frequency-limited gaps in the spectrum, which in turn may lead to audible artefacts.

Apart from these time- and frequency-limited gaps that may result from speech enhancement algorithms, additional time- or frequency-limited gaps may occur as a result of the transmission of speech over a communication channel. These can have different causes. On the one hand, the bandwidth of the transmitted speech is typically limited and thus only parts of the speech's lower frequencies are transmitted. On the other hand, errors can occur during the transmission of speech over, e.g., wireless networks, which lead to packet losses and thus missing intervals in the transmitted data. Both cases can be described by a binary spectral weighting by setting those entries of G to 0 where either frequencies or time intervals are missing.

Regardless of their exact shape, these gaps generally lead to a degraded speech quality and intelligibility. Therefore, already extensively researched approaches exist to reconstruct the speech signal in these gaps. For example, in [2], [3] different algorithms for *artificial bandwidth extension* (BWE) and in [4], [5] for *packet loss concealment* (PLC) are presented. Furthermore, there are also several studies on the reconstruction of temporal gaps in non-speech audio signals, e.g., [6], [7]. However, in most cases these algorithms are designed for a special class of problems and thus restricted to purely time- or frequency-limited gaps. For time- *and* frequency-limited gaps, e.g. resulting from a spectral weighting within a noise reduction as described above, those algorithms are usually not suitable. Only a few approaches, like in [8]–[10], already exist that are able to address this more generalized problem which is decoupled from a concrete application and is rather a solution to different types of interference that can occur in a wide variety of speech processing systems. Similar to *image inpainting* from digital image processing, for the case of speech signals these algorithms can be summarized under the term *speech inpainting*. While for a certain problem class a certain error type and thus also a certain corruption mask is predefined, the goal of speech inpainting is to be generally applicable to a wide range of problem classes. That is, speech inpainting is not intended to correct specific corruption patterns, but rather to learn some kind of internal models for the speech.

In this study, we propose to learn the complex mapping function from corrupted to uncorrupted speech using nonlinear *deep neural network* (DNN)-based regression models. We do this by first transforming the signals into a compressed representation given by the *mel frequency cepstral coefficients* (MFCCs) instead of training the DNNs directly on, e.g., the *short-time Fourier transform* (STFT). This reduces the DNNs' complexity and leads to significantly shorter training times. While previous studies often only focused on the reconstruction of specific gaps, we propose a statistically based error model with the target to cover a broad range of different shaped gaps. Specifically, we propose to train on randomly distributed rectangular gaps to obtain DNNs that are applicable to a variety of different specific problems without the need for re-training. Thus, in this study, we evaluate these DNNs not only on such randomly distributed gaps, but also on gaps that occur in the

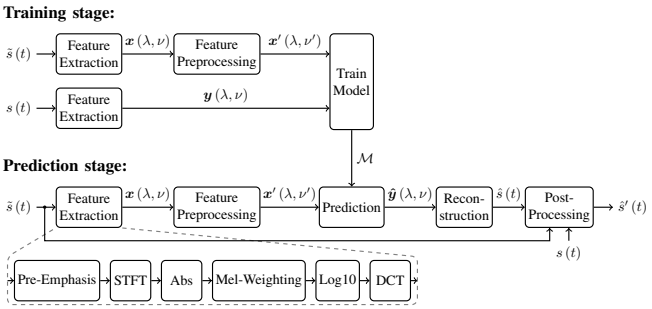


Fig. 1. Block diagram of the proposed speech inpainting algorithm.

specific problem of BWE. According to our knowledge, this is one of the first research that has trained DNNs in this way and investigated their generalizability to specific problems from speech processing.

II. Speech Inpainting Algorithm

Fig. 1 shows the speech inpainting algorithm. In the training stage, a regression DNN model is trained using pairs of uncorrupted and corrupted speech sequences s and $\tilde{s}$, respectively. In the prediction stage, the trained model $\mathcal{M}$ is used to reconstruct the predicted uncorrupted speech sequence $\hat{s}$ based on a corrupted speech sequence $\tilde{s}$.[1] The individual elements of the block diagram are explained in detail in the following subsections.

A. Sequence Creation

For the creation of the uncorrupted and corrupted sequences used for the training, all $i = 1, \ldots, M$ individual speech signal files are first normalized to $-26\,\mathrm{dBoV}$ and resampled to $f_s = 16\,\mathrm{kHz}$. In order to prevent the DNN from merely learning to reconstruct silence, speech pauses at the beginning and end of the speech signals are cut off. The resulting normalized and trimmed speech signals s_i are concatenated and build up the uncorrupted sequence s. The corrupted sequence $\tilde{s}$ is generated by concatenating the corrupted signals $\tilde{s}_i$. These $\tilde{s}_i$ are created by multiplying the STFT spectra $S_i(\lambda, \mu)$ of the uncorrupted signals s_i with binary corruption masks $G_i(\lambda, \mu) \in \{0, 1\}$ and then calculating the inverse STFT (ISTFT), i.e.

$$\tilde{s}_i(t) = \mathrm{STFT}^{-1}\left(G_i(\lambda, \mu) \cdot S_i(\lambda, \mu)\right) \tag{1}$$

where the TF plane is spanned by the frame index $\lambda = 1, \ldots, \lambda_{i,\max}$ and the frequency bin index $\mu = 1, \ldots, \mu_{\max}$. Here, we use a 640 samples square-root hann window with 320 samples overlap and $\mu_{\max} = 640$ frequency bins for the STFT. The ISTFT is calculated by employing the same window for the overlap-add synthesis and then truncating the resulting signal back to the original length of s_i. Although $\lambda_{i,\max}$ depends on the length of s_i, the index i is omitted in the following for the sake of simplicity.

As already mentioned in the introduction, the corruption masks G_i take on different forms depending on the problem class, e.g. BWE, PLC etc. In this study, we propose to use a statistically based error model that generates random rectangular gaps for training in order to achieve a generally applicable solution to these problem classes. These random rectangular gaps are referred to as "uniform gaps" in the following and are uniformly distributed over both time and frequency. Besides the position of their centre, also the duration $\Delta\lambda$ and frequency width $\Delta\mu$ of each individual gap are randomly drawn from a uniform distribution, i.e. $\Delta\lambda \sim \mathcal{U}(\Delta\lambda_{\min}, \Delta\lambda_{\max})$ and $\Delta\mu \sim \mathcal{U}(\Delta\mu_{\min}, \Delta\mu_{\max})$. Since the gaps are random, the corruptness before the creation of training data can only be specified using a heuristic measure for the corruption percentage

$$\hat{p} = \frac{2N \cdot \overline{\Delta\lambda} \cdot \overline{\Delta\mu}}{\lambda_{\max} \cdot \mu_{\max}} \tag{2}$$

with $N \in \mathbb{N}_0$ being the number of inserted gaps, $\overline{\Delta\lambda} = \Delta\lambda_{\min} + (\Delta\lambda_{\max} - \Delta\lambda_{\min})/2$ being the mean duration and $\overline{\Delta\mu} = \Delta\mu_{\min} + (\Delta\mu_{\max} - \Delta\mu_{\min})/2$ being the mean frequency width of the individual gaps. Because of possible overlaps and gaps ranging out of the TF plane, this heuristic measure typically differs from the actual corruption percentage p given by the percentage of zeros contained in G_i. Nevertheless, the estimation in (2) can be considered as a rough upper bound for the corruption percentage and thus be used to determine the required number of gaps N for a desired $\hat{p}$.

B. Feature Extraction

In order to reduce the complexity, the created sequences s and $\tilde{s}$ are transformed into a lower-dimensional feature domain. This removes redundancy and leads to a faster training of the DNNs. Therefore, a feature extraction is used to transform the corrupted sequence $\tilde{s}$ into the input feature vectors $\boldsymbol{x}$ which serve as input for the DNNs. The output of the DNNs is defined by the output features $\boldsymbol{y}$ that are used as labels during training and are determined by applying a feature extraction to the uncorrupted sequence s. Since the predicted output features $\hat{\boldsymbol{y}}$ are used to reconstruct the predicted uncorrupted sequence $\hat{s}$ in the prediction stage, the total achievable performance is limited by the selected output features $\boldsymbol{y}$.

In this study, we selected the MFCCs as both input and output features. Specifically, we have used the implementation from [11] with the following parameters: 640 samples window length, 320 samples overlap, 128 spectral bands with a frequency band range from 20 Hz to 8 kHz, 32 cepstral coefficients, a preemphasis factor of 0.97 and a liftering exponent of 0.6. This number of cepstral coefficients was chosen to achieve a good tradeoff between complexity and achievable performance, reducing the amount of data per frame from 640 samples to only $\nu_{\max} = 32$ MFCCs.

C. Feature Preprocessing and DNN Training

The preprocessing is done by first normalizing the input features $\boldsymbol{x}$ to zero-mean and unit-variance. Then, each input frame λ is expanded by copies of it's $\tau = (T-1)/2$ preceding and succeeding frames where $T \in 2\mathbb{N} - 1$ is the input span.

[1] Although the corrupted sequence $\tilde{s}$ and thus all other variables in the training and prediction stage are not necessarily identical, additional indices are not included in this study for the sake of simplicity.

Thus, in total T adjacent frames are used as input to the DNN, giving it additional acoustic contextual information for inpainting the centered frame. After feature preprocessing, pairs of the resulting preprocessed input MFCCs $\boldsymbol{x}'$ and the output MFCCs $\boldsymbol{y}$ are shuffled and used for training the multiple output regression DNN.

D. Reconstruction and Post-Processing

Even though MFCCs were not originally developed for speech synthesis, different investigations on the inversion of MFCCs for speech enhancement already exist (see [12]). Here, we use the inversion algorithm given in [11] to reconstruct the predicted uncorrupted sequence $\hat{s}$ from the predicted output MFCCs $\hat{\boldsymbol{y}}$. Since the MFCCs discard the phase information of the original signal, this implementation uses white noise as excitation for the reconstruction.

After reconstruction, additional post-processing steps can be applied to the predicted uncorrupted sequence $\hat{s}$ in order to generate the actual inpainted sequence $\hat{s}'$. In typical applications, the corruption mask G, i.e. the position and shape of the gaps, is known or can be easily detected. This knowledge is exploited in our first post-processing step. Therefore, only the gaps in the STFT of the corrupted sequence $\tilde{s}$ are filled with the corresponding STFT parts of the predicted sequence $\hat{s}$, where the gaps are determined by comparing the STFTs of s and $\tilde{s}$ ("filled gaps"). For evaluation purposes, another optional post-processing step can be applied in order to eliminate the phase error due to the white noise excitation. This is done by replacing the phase of the predicted uncorrupted sequence $\hat{s}$ with the phase of the original uncorrupted sequence s ("cheated phase").

III. Experimental Results and Analysis

All experiments were performed on the VCTK database which is an English speech corpus comprising several speakers with different accents [13]. We splitted the speakers into three subsets, namely training, validation and test, so that all subsets contain approximately the same proportion of male and female speakers. After sequence creation (see Sec. II-A), this results in approximately 18.5 hours of training, 3.5 hours of validation and 3.5 hours of test data. Similar to [2], [14] we selected three fully connected feedforward neural networks (NNs) with only $N_\mathrm{L} = 1$ hidden layer and different hidden units per layer $N_\mathrm{U} = 512, 2048, 6144$ as well as two DNNs with $N_\mathrm{L} = 2, 3$ hidden layers and $N_\mathrm{U} = 2048$ hidden units per layer (denoted as $\mathrm{DNN}_{N_\mathrm{U}}^{N_\mathrm{L}}$). These different DNN configurations were each trained for different input spans $T = 1, 3, 5, 7, 9, 11$. We trained all DNNs for 50 epochs using a batch size of 128 frames, batch normalization after each hidden layer, rectified linear units (ReLUs) as activation functions for the hidden layers and linear functions for the output layer. Xavier uniform was used as initializer [15], AdaGrad as optimizer [16] and the mean squared error (MSE) as loss function. After each full epoch, the DNNs were evaluated on the validation set, so that the DNN at the most powerful epoch can be selected at the end of the training.

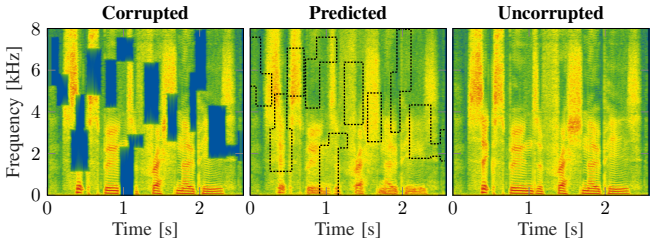


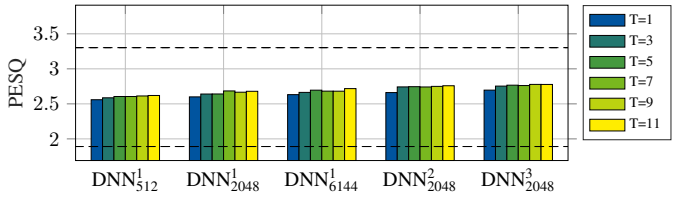
Fig. 2. Exemplary speech inpainting result for the uniform case. From left to right: Corrupted, predicted (with highlighted gaps) and uncorrupted STFT.

Same as for the loss function, we also used the MSE of the output MFCCs to determine the best epoch. Three instrumental measures were chosen for evaluating the quality of the inpainted sequence $\hat{s}'$. Those are *log-spectral distance* (LSD in dB), *short-time objective intelligibility* (STOI) [17] and *perceptual evaluation of speech quality* (PESQ) [18]. For all three measures, the uncorrupted sequence s is always used as reference in order to evaluate the overall performance including all processing steps. In the following experiments, only the most relevant results of the quality measures and the examined DNN configurations are given due to space limitation and for comprehensibility.

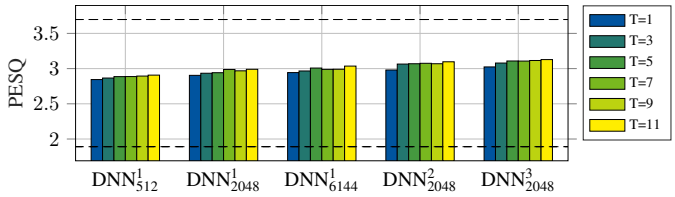
A. Evaluation for the Uniform Case

Before investigating their generalizability (see Sec. III-B), we first evaluate the DNNs trained on uniform gaps for uniform gaps as well. Therefore, we created the training and validation data with uniform gaps using a duration ranging from $\Delta\lambda_\mathrm{min} = 5$ (i.e. 100 ms) to $\Delta\lambda_\mathrm{max} = 15$ (i.e. 300 ms), a frequency width ranging from $\Delta\mu_\mathrm{min} = 20$ (i.e. 500 Hz) to $\Delta\mu_\mathrm{max} = 120$ (i.e. 3 kHz) and a heuristic corruption percentage of $\hat{p} = 30\%$ (see Sec. II-A). An exemplary speech inpainting result for this case can be seen in Fig. 2. More audio samples are available online (see [19]). As can be seen from the corrupted STFT, there is almost no time frame (i.e. column) that does not contain any corruption. Conventional schemes like a PLC algorithm under frame erasure condition, i.e. erasing all corrupted frames, would not have enough uncorrupted context and would fail to reconstruct this kind of data. Hence, they are not suitable for a fair comparison. In contrast, our algorithm can be applied to arbitrary corruptions which highlights the flexibility of our proposed speech inpainting algorithm.

First, only the fill gaps step is used after all reconstructions, including those for the reference scores. Fig. 3a shows the average PESQ results on the uniform validation set for different DNN configurations $\mathrm{DNN}_{N_\mathrm{U}}^{N_\mathrm{L}}$ and different input spans T. The horizontal black dashed lines in Fig. 3a and 3b serve as lower and upper bounds for the trained DNNs, showing the limitations of quality due to the lossy MFCC transformation. They represent the case where the DNNs predict the labels without any error (upper line) or where no inpainting is done at all (lower line). Specifically, the lower line results from a comparison with the original corrupted sequence $\tilde{s}$ and the upper line is the PESQ value of the sequences reconstructed from the uncorrupted MFCCs $\boldsymbol{y}$, which result from the feature extraction applied to the uncorrupted sequence s. It can be



(a) Using white noise as excitation.



(b) Using original phase spectrum (i.e. cheated phase).

Fig. 3. Average PESQ results with different input spans T on the uniform validation set for different DNN configurations (ascending sorted by complexity). Only gaps in the TF plane are filled using the prediction from the DNNs. The horizontal lines represent upper and lower reference quality limits.

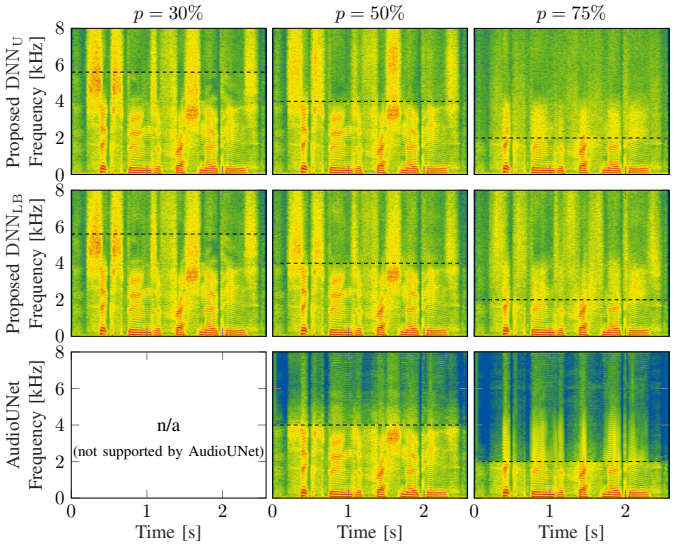


Fig. 4. Exemplary speech inpainting results for the lower-bandwidth case using filled gaps sequence with different models. Top: $\mathrm{DNN_U}$ trained on uniform gaps. Middle/Bottom: $\mathrm{DNN_{LB}}$ and waveform-based AudioUNet specifically trained on the individual lower-bandwidth data. The horizontal lines mark the cut-off frequencies f_c. The corresponding uncorrupted STFT can be found in Fig. 2.

seen that the PESQ values of all trained DNNs exceed at least half of the reference range and hence drastically increase the performance compared to no modification. Furthermore, it can be seen in Fig. 3a that the PESQ value is improving for increasingly complex DNNs. Especially the increasing number of layers N_L has a positive influence. As became clear from further investigations, this is mainly due to the batch normalization, which is particularly helpful for the learning process of deeper DNNs. Similar to the results in [14], increasing the input span T (at least up to $T = 11$) tends to improve the PESQ value for a fixed DNN configuration.

Now, as an alternative for the white noise excitation, we investigate the cheated phase case. This allows us to examine the extent our algorithm can be improved with a more sophisticated phase reconstruction technique. Fig. 3b shows the new average PESQ results on the validation set when the post-processing step for cheating the phase is added. Again, black dashed reference lines are drawn in. While the lower line is equivalent to Fig. 3a, the upper line increased by 0.39. As expected, the performance of all DNN configurations also improved. Specifically, the average PESQ score increases by 0.31, which is roughly equivalent to the increase of the upper line. This shows that even better results can be expected with a more complex phase reconstruction technique, whose lower and upper bounds are approximately given by the results in Fig. 3a and Fig. 3b.

B. Evaluation for the Lower-Bandwidth Case

Now, we investigate whether the proposed way of training on randomly distributed rectangular gaps leads to DNNs that are applicable to other specific problems from speech processing. Specifically, in this study, we consider the BWE for this purpose. Therefore, we have generated three new training and validation sets, which do not have uniform gaps, but instead bandwidths reduced by the corruption percentages $p = 30\%, 50\%, 75\%$. Since we have always removed the upper frequencies, i.e. from the nyquist frequency $f_s/2 = 8\,\mathrm{kHz}$ downwards, this results in lower-bandwidth data with cut-off frequencies $f_c = 5.6\,\mathrm{kHz}, 4\,\mathrm{kHz}, 2\,\mathrm{kHz}$. For this investigation we selected the best performing DNN configuration from the uniform case, i.e. $N_\mathrm{L} = 3$ hidden layers and $N_\mathrm{U} = 2048$ hidden units per layer with an input span $T = 11$.

For each corruption percentage p, we trained a DNN on the individual lower-bandwidth data, which we summarize as $\mathrm{DNN_{LB}}$ in the following. The best DNN presented in Sec. III-A was intentionally not re-trained on this new lower-bandwidth data and is denoted as $\mathrm{DNN_U}$ in the following. As in the previous section, we again use the fill gaps sequence for all evaluations. In order to compare our results with a state-of-the-art algorithm, we use the AudioUNet [3] as one more reference and therefore trained it on the individual lower-bandwidth data. In contrast to the proposed $\mathrm{DNN_U}$ and $\mathrm{DNN_{LB}}$, the AudioUNet is working on raw waveform audio data. In Fig. 4 exemplary speech inpainting results of all three models for the lower-bandwidth case are depicted. More audio samples are available online (see [19]). While the AudioUNet is only applicable for corruption percentages corresponding to upscaling factors of 2, 4, 6, etc., our algorithm can be applied to any corruption percentage. This, again, highlights the flexibility of our proposed speech inpainting algorithm.

Table I shows the results for the three models. While $\mathrm{DNN_U}$ was only trained on uniform gaps, $\mathrm{DNN_{LB}}$ and AudioUNet were specially trained on the different lower-bandwidth data percentages. Our $\mathrm{DNN_U}$ and $\mathrm{DNN_{LB}}$ outperform the AudioUNet w.r.t. LSD for all corruption percentages p when using white noise (WN) excitation. In terms of STOI, our DNNs almost achieve identical results as the AudioUNet and $\mathrm{DNN_{LB}}$ even surpasses it for $p = 75\%$. Thus, the results of $\mathrm{DNN_U}$ are very close to those of $\mathrm{DNN_{LB}}$ and the AudioUNet, even though $\mathrm{DNN_U}$ has never been trained for

TABLE I
AVERAGE LSD/STOI RESULTS ON THE LOWER-BANDWIDTH DATA WITH DIFFERENT CORRUPTION PERCENTAGES p. WN DENOTES THE CASE WHERE WHITE NOISE IS USED AS EXCITATION AND CP DENOTES THE CHEATED PHASE CASE.

		$p = 30\%$ $f_c = 5.6\,\mathrm{kHz}$		$p = 50\%$ $f_c = 4\,\mathrm{kHz}$		$p = 75\%$ $f_c = 2\,\mathrm{kHz}$	
		LSD	STOI	LSD	STOI	LSD	STOI
$\mathrm{DNN_U}$	WN	3.072	0.9999	4.736	0.9982	7.935	0.9126
	CP	2.321	0.9999	3.787	0.9987	6.761	0.9260
$\mathrm{DNN_{LB}}$	WN	3.066	0.9999	4.502	0.9984	6.644	0.9408
	CP	2.230	0.9999	3.470	0.9989	5.393	0.9529
AudioUNet		n/a	n/a	6.105	0.9985	8.688	0.9185

this special problem of BWE. This confirms our hypothesis that training DNNs on randomly distributed rectangular gaps leads to a robust solution which is applicable for various problems from speech processing. To even further validate this hypothesis, we investigated the cross check, i.e., how $\mathrm{DNN_{LB}}$ behaves on the uniform gaps. Specifically, we applied $\mathrm{DNN_{LB}}$ trained for $p = 50\%$ to the same sequences as in Sec. III-A and obtained PESQ scores of 1.66 and 1.74 for white noise excitation and the cheated phase case, respectively. Both scores are even below the lower bound of 1.9, i.e., when the original corrupted sequence is used. This shows that while $\mathrm{DNN_{LB}}$ is not applicable to problems other than BWE, $\mathrm{DNN_U}$ trained on uniform gaps is very flexible and can be applied to different problem classes.

Considering the cheated phase (CP) case, our proposed $\mathrm{DNN_U}$ and $\mathrm{DNN_{LB}}$ surpass the AudioUNet for all corruption percentages w.r.t. both LSD and STOI. Accordingly, even better results can be expected when a more sophisticated phase reconstruction technique is used. Moreover, this demonstrates that our proposed algorithm is able to outperform state-of-the-art algorithms when trained for a specific problem. The good results of our algorithm suggest that MFCCs are particularly suitable for prediction of the upper frequencies. This is probably due to the fact that at high frequencies a rough, rather noisy estimate of the speech's STFT is sufficient. Here the MFCCs benefit from the fact that they smooth this upper frequency range very strongly due to the mel-weighting and a low frequency resolution in this range. In the lower frequencies, on the other hand, speech typically contains harmonic structures that are attenuated by smoothing during the MFCC transformation.

IV. CONCLUSIONS

In this work, a speech inpainting algorithm for the reconstruction of missing parts in the TF representation of speech using DNNs trained on MFCCs is proposed. Overall, a significant improvement of the PESQ score from 1.9 up to 2.8 can be achieved for speech inpainting on random rectangular gaps where conventional schemes like a PLC algorithm can not be applied successfully. On average, the use of more acoustic contextual information by increasing the input span leads to an improvement in performance, i.e. the achieved PESQ score. We have found that the MFCCs are particularly helpful for reconstruction of the upper frequencies of speech. Our target was to train DNNs that generalize and are applicable to different use cases from speech processing. Thus, we did not focus on the network architecture, but on the training data. We showed that our proposed way of training DNNs on random rectangular gaps leads to an algorithm that, applied to BWE, achieves almost the same performance level of specially trained DNNs. In the future, we intend to further investigate this generalizability to other specific problems, e.g., PLC.

Acknowledgment: Simulations were performed with computing resources granted by RWTH Aachen University under project rwth0474.

REFERENCES

[1] D. Wang and J. Chen, "Supervised speech separation based on deep learning: An overview," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 10, pp. 1702–1726, 2018.

[2] K. Li and C. Lee, "A deep neural network approach to speech bandwidth expansion," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 4395–4399.

[3] V. Kuleshov, S. Z. Enam, and S. Ermon, "Audio super resolution using neural networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[4] B. Lee and J. Chang, "Packet loss concealment based on deep neural networks for digital speech transmission," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 2, pp. 378–387, Feb. 2016.

[5] P. Prablanc, A. Ozerov, N. Q. K. Duong, and P. Pérez, "Text-informed speech inpainting via voice conversion," in *24th European Signal Processing Conference (EUSIPCO)*, Aug. 2016, pp. 878–882.

[6] Y.-L. Chang, K.-Y. Lee, P.-Y. Wu, H. yi Lee, and W. Hsu, "Deep long audio inpainting," 2019, arXiv:1911.06476.

[7] P. P. Ebner and A. Eltelt, "Audio inpainting with generative adversarial network," 2020, arXiv:2003.07704.

[8] B. R. Ramakrishnan, "Reconstruction of incomplete spectrograms for robust speech recognition," Ph.D. dissertation, Carnegie Mellon University, Apr. 2000.

[9] S. Thakallapalli, S. Gangashetty, and N. Madhu, "A new weighted NMF algorithm for missing data interpolation and its application to speech enhancement," in *27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[10] M. Kegler, P. Beckmann, and M. Cernak, "Deep speech inpainting of time-frequency masks," in *Interspeech*, Oct 2020.

[11] D. P. W. Ellis, "PLP and RASTA (and MFCC, and inversion) in Matlab," 2005, online web resource. [Online]. Available: http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/

[12] L. E. Boucheron and P. L. D. Leon, "On the inversion of mel-frequency cepstral coefficients for speech enhancement applications," in *International Conference on Signals and Electronic Systems*, Sep. 2008, pp. 485–488.

[13] C. Veaux, J. Yamagishi, and K. MacDonald, "CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit," 2017.

[14] Y. Xu, J. Du, L. Dai, and C. Lee, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal Processing Letters*, vol. 21, no. 1, pp. 65–68, Jan. 2014.

[15] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. AISTATS*, 2010, pp. 249–256.

[16] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[17] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2010, pp. 4214–4217.

[18] ITU, "Rec. p.862.2: Wideband extension to recommendation p.862 for the assessment of wideband telephone networks and speech codecs," 2018.

[19] L. Thieling and P. Jax, "Generally applicable deep speech inpainting using the example of bandwidth extension | listening examples," 2021, online web resource. [Online]. Available: https://www.iks.rwth-aachen.de/qr/eusipco2021-si