Evaluating the Effect of Noise Suppression on the Performance of an Automatic Speech Recognition System Trained End-to-End

by Tommy Fox

Submitted in partial fulfillment of the requirements for the
Master of Music in Music Technology
in the Department of Performing Arts Professions
Steinhardt School
New York University

Advisor: Dr. Brian McFee
Date: December 2, 2019

Abstract

Reliable automatic transcription of speech signals in noisy environments enhances technological access for individuals whose interaction would be otherwise limited. Applications include automatic closed captioning of video, hands-free voice control, and language translation systems among many others. Recently, end-to-end neural networks have been employed for the noisy speech transcription task, requiring little to no pre-processing of input data. In lieu of arduous feature engineering found in classical systems, end-to-end networks require large volumes of training data and carefully designed network architecture instead. While end-to-end neural networks have shown state of the art accuracy in transcribing noisy speech, there remains room for improvement. This work examined the effect of source separation preprocessing on the accuracy of the Deep Speech network, which was trained end-to-end. The data used for this experiment included two clean speech datasets and one dataset consisting of noise commonly found in urban environments. Deep Speech's accuracy was tested using clean speech, unseparated noisy speech, and finally noisy speech after separation. Source separation methods included Open-Unmix, which utilizes Recurrent Neural Network-based Bidirectional Long-Short Term Memory, as well as Non-negative Matrix Factorization. Deep Speech's Word Error Rate consistently improved across all signal to noise ratios and datasets processed with Open-Unmix, while Non-negative Matrix Factorization processing provided negligible improvement. These results indicate that deep learning context-based source separation preprocessing for speech transcription is an avenue worthy of further research, as is improving the model robustness of stand-alone end-to-end speech recognition systems.

Table of Contents

Appendices

List of Figures

3. Procedure

4. Results and Discussion

Appendix A

Appendix B

List of Abbreviations

ASR	Automatic Speech Recognition
BSS	Blind Source Separation
CNN	Convolutional Neural Network
CSV	Comma-separated Value
CTC	Connectionist Temporal Classification
DNN	Deep Neural Network
DFT	Discrete Fourier Transform
GAN	Generative Adversarial Network
GMM	Gaussian Mixture Model
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
ISTFT	Inverse Short-time Fourier Transform
LPC	Linear Predictive Coefficient
LSTM	Long Short-term Memory
LUFS	Loudness Units relative to Full Scale
MFCC	Mel-frequency Cepstral Coefficient
NMF	Non-negative Matrix Factorization
RNN	Recurrent Neural Network
SAR	Signal to Artifact Ratio
SDR	Signal to Distortion Ratio
SIR	Signal to Interference Ratio
SNR	Signal to Noise Ratio
STFT	Short-time Fourier Transform
UMX	Open-Unmix Source Separation Method
WER	Word Error Rate

1. Introduction and Motivation

Automatic Speech Recognition (ASR) is a heavily researched subject area and is the underlying technology of many applications, including verbal human-computer interaction and automatic closed captioning, among others. ASR has the power to augment the accessibility of modern technology for human use. For example, YouTube utilizes noise robust ASR to caption user video uploads, making content more accessible to the hearing impaired (Soltau, Liao, Sak, 2017). Additionally, ASR enables hands-free computer interaction, the benefits of which are found while operating a vehicle, machinery, or by those with physical impairments. ASR is also a staple in oral translation systems, which are especially useful in linguistically rich regions of the world (Radha, Vimala, 2012). This is far from an exhaustive list of ASR's potential uses and benefits. The task can be approached numerous ways depending on the application, but ultimately seeks the same goal in all settings: accurately transcribing the user's acoustic speech signal.

Regardless of the application, a persistent hurdle for ASR accuracy is transcribing noisy speech signals. The degree of noise corruption can vary widely and highly depends upon the setting in which the system is deployed. For instance, when automatically captioning YouTube videos, speech may be entangled with unpredictable sources of noise, calling for a sophisticated noise reduction process prior to speech transcription. Other circumstances, such as orally controlling a device while driving, may encompass more foreseeable noise patterns, which can inform the noise removal and transcription process. In general, noise suppression preprocessing has been found to improve the accuracy of ASR systems in noisy environments.

While classical ASR systems benefit from preprocessing stages regardless of their application to noisy or clean speech, recent successful implementations of end-to-end neural networks have enabled researchers to side step preprocessing altogether. End-to-end neural networks harness the modern resources of computational power in tandem with vast amounts of training data. These networks allow for the encoding and decoding of raw input signals, as is consistent with the motivation for their development. While these systems can provide state of the art accuracy in learning complex nonlinear functions, there remains room for improvement. In this regard, little attention has been given to the effects of utilizing preprocessed data in end-to-end ASR systems, particularly noise suppression. End-to-end ASR systems may be further exploited by utilizing noise suppression preprocessing along with the aforementioned modern resources. The possibility that noisy end-to-end ASR improvement may come, in part by way of noise suppression preprocessing is worthy of investigation. This research provides insight on the effect of providing end-to-end neural networks with preprocessed data through the lens of noisy speech transcription, while implicitly testing the model robustness of an end-to-end ASR system.

2. Literature Review

2.1 Historical Background

Automatic Speech Recognition has a long and rich history of research dating back more than 60 years (Li, Deng, Haeb-Umbach, Gong, 2015). The first ASR system, created at Bell Labs in 1952, was designed to recognize spoken digits (Washani, Sharma, 2015). The general problem addressed in ASR systems is converting an acoustic speech utterance into text and/or classifying the identity of the speaker (Washani, Sharma, 2015). Original approaches of ASR systems were based on hard-coded templates of speech signals, which compared a given acoustic utterance to a database of pre-recorded speech signals (Radha, Vimala, 2012). While an impressive feat at the time, this type of system was computationally expensive and not robust to speaker variance (Radha, Vimala, 2012). Further research led to the use of statistical-based approaches, which endured for a significant span of ASR research history and remained state of the art until about 2009 (Li, et al., 2015). For about the past ten years to present, state of the art systems have been rooted in deep neural networks, a technology fostered by the age of information and increasing computing capabilities at continuously decreasing cost.

The ASR task can generally be segmented into two categories: classifying an utterance with varying vocabulary size, and classifying speaker identity (Radha, Vimala, 2012). Classifying utterances can be further broken into scenarios in which a person vocalizes a single word, a few words, a sentence, or more (Radha, Vimala, 2012). Single words, or short phrases with silence occurring before and after the utterance are relatively easier to classify than continuous speech; finding an appropriate segmentation of the signal is a particular challenge

when words run together (Radha, Vimala, 2012). While this problem is absent in the keyword detection portion of personal assistant systems like Apple's Siri, where the goal is to detect the wake-up phrase "Hey Siri" (Li, et al. 2015), short utterances are not typical of natural speech patterns. The most persistent challenges for ASR systems include continuous speech, speaker variation, and background noise (Washani, Sharma, 2015). The predominant performance evaluation of ASR systems is the Word Error Rate (WER), which indicates the ratio of inaccurate text predicted by the system to the ground truth transcription at the word level.

ASR systems are broken into two segments: front-end and back-end. The front-end of the system extracts and encodes acoustic features, while the back-end, usually termed a language model, accounts for language and grammar considerations (Radha, Vimala, 2012). The language model outputs the most plausible text sequence, given an encoded acoustic input provided by the front end. For example, the front end may encode an utterance as "I wint to bostin.", where the language model decodes the sequence as, "I went to Boston." (Hannun, Case, Casper, Catanzaro, Diamos, Elsen, …, Ng, 2014). While statistical methods are still commonplace in language modeling, deep learning approaches have recently shown promising results (Washani, Sharma, 2015). Over time, the front-end of ASR systems has significantly evolved and improved. A general consensus is that the quality of feature extraction and acoustic model encoding are the most salient contributors to accuracy in ASR systems, while not diminishing the importance of language modeling. Ultimately, end-to-end ASR approaches seek to learn the language model from the acoustic utterances themselves and eliminate back-end processing altogether (Graves, Jaitly, 2014).

2.2 Feature Extraction

Fundamental preprocessing methods for speech recognition include framing, windowing, end point detection, pre-emphasis filtering, and noise filtering. During preprocessing, classical systems seek to segment speech into phonemes (Li, et al., 2015). One main challenge posed in this approach is that the same utterance can have numerous lengths depending on who is speaking and how (Li, et al., 2015). A common algorithm for dealing with this is dynamic time warping, which measures the similarity between two sequences that vary in time, or speed (Radha, Vimala, 2012).

One approach of classical ASR systems is segmenting a speech signal into analysis windows, and determining whether the signal resembles a consonant or a vowel, which informs the rest of the encoding process (Washani, Sharma, 2015). Spoken consonants resemble pure noise signals and are classified as unvoiced, while vowels display periodic, pitched behavior and are classified as voiced (Park, 2009). Popular methods for determining voiced and unvoiced signals include zero crossing rate (Park, 2009), autocorrelation, and cepstral peak analysis (Washani, Sharma, 2015). Cepstral analysis methods operate upon frequency domain signals in ways that are typical of the time domain (Park, 2009).

The most common acoustic features used to create acoustic speech models are Linear Predictive Coefficients (LPC) and Mel-frequency Cepstral Coefficients (MFCC). LPCs approximate a speech signal as a linear combination of past samples and seek to minimize the sum of squared differences between observed speech samples and predicted values (Radha, Vimala, 2012). A common approach to computing LPCs utilizes autocorrelation (Park, 2009).

MFCCs can generally be thought of as a low frequency accentuation process, which extracts the spectral shape of a signal. The resulting scaled frequency components resemble the model for human sound perception. MFCCs are obtained by first computing a Mel-scaled power spectrum of the input, which concentrates low frequency energy. The Mel-scaled power spectrum is then compressed using the logarithm operator, and finally processed with the discrete cosine transform, a technique commonly used in image compression (Radha, Vimala, 2012). These features tend to be invariant to characteristics not informative to audio classification.

2.3 Acoustic Model Encoding

Statistical approaches to acoustic modeling were introduced in the 1970's and remained state of the art until about 10 years ago (Li, et al., 2015). One statistical method employs a Hidden Markov Model (HMM) combined with a Gaussian Mixture Model (GMM) to represent the temporal evolution of a speech signal. In this process, speech variations are modeled using the automatic learning procedure of an HMM trained with a large amount of pre-recorded speech data (Radha, Vimala, 2012). HMMs are a context dependent decision process, where the likelihood of the state of a current sample x[t] depends upon the state of the previous sample x[t-1], where x is an input signal and t refers to a discrete sample in time. These states are informed by GMMs, trained to represent the statistical characteristics of speech (Li, et al., 2015). HMMs represent complete words and sequences from phoneme information by optimizing a network search for the optimal word sequence (Radha, Vimala, 2012). The simplicity and efficiency of maximum likelihood estimation algorithms contributed to the rise and endurance

of HMM-GMM acoustic modeling in classical ASR systems (Li, et al., 2015).

Neural network acoustic models and other connectionist approaches were first introduced into speech pipelines in the early 1990s (Hannun, et al., 2014), however the amount of data and computational power required to train such models rendered them impractical until relatively recently. Around 2009, researchers at Microsoft and the University of Toronto made a major push to alleviate the shortcomings of statistical models through the use of deep neural networks (DNN) (Dahl, Yu, Deng, Acero, 2011). The advent of applying DNNs to ASR systems represents the most important of recent developments in the task (Li, et al., 2015). DNNs vastly outperform statistical methods in front-end ASR encoding, however in order to maintain accuracy, they require a relatively large corpus of training data as well as nontrivial computing power (Li, et al., 2015). Though bound by eventual physical limitations, Moore's Law states that the cost of computing power will continue to decrease while resources increase on a yearly basis, contributing to DNNs becoming an increasingly practical tool over time (Li, et al., 2015). The repurposing of Graphics Processing Units (GPUs) for deep learning has significantly contributed to its surge. Currently, all major commercial ASR systems are based on deep learning methods (Li, et al., 2015).

DNNs model an acoustic signal by learning complex nonlinear functions given training data with optimization carried out through gradient descent (Li, et al., 2015). Deep learning is able to model high level abstractions in data, using non-linear transformations, the most common being the Hyperbolic Tangent, Sigmoid, and Rectified Linear Unit functions (Li, et al., 2015).

In the classical neural network approach, predictions indicate the likelihood that a certain phoneme, letter, or word is present within a given signal frame (Li, et al., 2015). Modern DNN-based systems utilizing Connectionist Temporal Classification (CTC) are able to make predictions given unaligned data (Graves, Jaitly, 2014).

A fundamental principle of deep learning's data-driven approach to problem solving is doing away with hand-crafted features and relying on raw input (Li, et al., 2015). However, if not given an adequate number of training examples, DNNs are prone to overfitting training data and not generalizing well to signals unseen during the training process. Feature extraction methods tend to alleviate this overfitting and improve performance. The most common and robust features used in DNN-based ASR systems are LPCs and MFCCs (Radha, Vimala, 2012).

Advances in the computer vision community have also impacted ASR systems. Convolutional Neural Networks (CNN), which were developed for image recognition, can treat an audio spectrogram as an image in order to classify sound (Li, et al., 2015). Typically, these networks process information with initial convolutional layers, which progressively detect higher level features in the image, proceeded by the fully connected layers of a rudimentary DNN (Li, et al., 2015). Various implementations of these networks have proven useful in addressing ASR, however relatively recent developments in deep learning have further improved performance.

Recurrent Neural Networks (RNNs) tend to be more effective than classical DNNs for ASR because the activation of each layer not only depends upon current input, but also previous (unidirectional) and sometimes future (bidirectional) activations (Li, et al., 2015). The construct

of memory in RNNs provides temporal context to the decision making process for a given frame, which is suitable for audio data and particularly powerful in ASR systems (Li, et al., 2015). Further, a memory mechanism called Long Short-term Memory (LSTM) accounts for including an arbitrary contextual length in the encoding process, providing a more robust and complex network architecture. LSTM-based systems are able to learn long time scale structure without losing the short term information provided by the base of the RNN structure (Horchreiter, Schmidhuber, 1997). While RNNs are currently the state of the art for the task of ASR, recent incorporation of LSTM into CNN architecture has shown promising results (Zhang, Chan, Jaitly, 2017). Performance of these systems suggests that context is an important factor in modeling speech signals and audio signals in general.

End-to-end speech recognition has long been a goal of the research community, wherein the neural network design implements a holistic learning process and preprocessing of input is minimal or absent altogether (Soltau, et al., 2017). These networks apply the gradient-based learning method to the system as a whole, where all modules are differentiable (Glasmachers, 2017). This approach seeks to remove the need for hand-crafted features and solve tasks in a purely data-driven manner (Glasmachers, 2017). End-to-end networks are complex and deep, with the main focus upon designing systems for training efficiency and architectures that best fit the tasks at hand (Glasmachers, 2017). This emphasis on network design and training data curation occurs in lieu of feature engineering.

There exist several approaches to designing end-to-end systems (Soltau, et al., 2017). Google researchers recently presented a system titled Neural Speech Recognizer, utilizing an

RNN with LSTM to encode word level targets. The system achieves competitive performance on the continuous speech task, while forgoing the use of a back-end language model (Soltau, et al., 2017). This network provides superior performance to phone-level modelling and is able to accurately transcribe song lyrics, although it was not trained with sung speech data (Soltau, et al., 2017).

The Deep Speech network is another end-to-end RNN-based system that performs well in classifying both conversational and noisy speech (Hannun, et al., 2014). Feature inputs to the network are simply the log-scaled power spectrum of an audio sample, and classification is broken down into individual letters, foregoing the phonetic approach of classical systems (Hannun, et al., 2014). As is common in end-to-end system design, emphasis was given to optimizing the network architecture to efficiently utilize GPUs for training with a large amount of data (Hannun, et al., 2014). The network is simpler than most end-to-end systems in that it implements a 5 layer neural network, with only 1 recurrent layer and does not use an LSTM module, which increases its speed during training as well as its scalability (Hannun, et al., 2014). The single recurrent layer is bidirectional and learns its weight and bias parameters from past, current, and future input samples (Hannun, et al., 2014); at each time t, output depends upon frame x[t], along with context of frames on either side, x[t-1] and x[t+1]. The back end of Deep Speech implements a 5-gram language model trained with 220 million phrases, constituting a 495,000 word vocabulary (Hannun, et al., 2014).

A combination of the Connectionist Temporal Classification (CTC) objective function and beam search post processing allows Deep Speech to transcribe without frame alignment

and without an HMM interface to the language model (Hannun, Maas, Jurafsky, Ng, 2014). The CTC output maximizes the likelihood of an output sequence by summing over all possible input-output sequence alignments, removing the iterative alignment process found in classical systems (Hannun, et al., 2014). CTC in combination with beam search enables the direct integration of a language model to the front-end of the system, and is more closely related to the final Word Error Rate evaluation metric (Graves, Jaitly, 2014). This inherently optimizes the logarithmic probability of state sequences rather than the likelihood of a single input as in classical systems (Sak, Senior, Rao, Irsoy, Graves, Beaufays, Schalkwyk, 2015). This also allows for speech classification at the character level, further removing expert phoneme knowledge from the task.

The creators of Deep Speech put significant effort into curating a large amount of training data, required for such large networks to perform well (Hannun, et al., 2014). The majority of Deep Speech's training data (5,100 hours) consists of unaccented English speech, wherein speakers were recorded while reading text aloud. Deep Speech's training data also consisted of 2,300 hours of conversational speech. In total Deep Speech was trained with about 7,400 hours of speech data (Hannun, et al., 2014). In 2016, Deep Speech 2 was introduced, expanding upon the original framework in terms of training efficiency and primarily exhibiting its ability to adapt to new languages, specifically Mandarin Chinese (Amodei, Ananthanarayanan, Anubhai, Bai, Battenberg, Case, …, Chen, 2016).

2.4 Noise Robust Automatic Speech Recognition

Despite a deep history of research spanning over 30 years (Li, et al., 2015), accuracy in noisy environments remains one of the most important ASR challenges (Zhang, Geiger, Pohjalainen, Mousa, Jin, Schuller, 2018). In this context, noise is defined as an additive or convolutional corruption of the speech signal. Additive noise can be classified into two categories: stationary noise, which is constant with respect to time, such as white noise, and non-stationary noise, such as music (Zhang, et al., 2018). Non-stationary noise can further be classified into environmental noise, consisting of sounds occurring in one's natural surroundings, and multi-talker noise, where the target speaker and at least one additional speaker talk at the same time. Multichannel convolutive reverberation can also be considered noise distortion (Li, et al., 2015) but exceeds the scope of this paper, where single channel additive noise is the focus. Compensating for non-stationary noise remains a challenging task because these noise signals can exhibit exceptionally complex nonlinear functions. Acoustic modeling of these noise signals is nontrivial. Approaches to solving the noisy ASR task mainly focus upon either obtaining a cleaner speech signal during preprocessing, or training the encoder to model noisy speech, foregoing noise suppression altogether.

As is consistent with the ASR task in general, classical methods of noise removal are rooted in statistical modeling of noise, while currently the most effective implementations for removing non-stationary noise utilize deep learning technology (Li, et al., 2015). MFCCs alone are considered a relatively noise robust feature, however the addition of noise removal preprocessing is widely beneficial to ASR systems (Li, et al., 2015). Statistical models are

efficient for removing stationary noise, where the distribution of a constant noise signal is obtained and then removed from the speech signal (Zhang, et al., 2018). These separation methods include spectral subtraction, wiener filtering (Zhang, et al., 2018), and non-negative matrix factorization (NMF) (Raj, Virtanen, Chaudhuri, Sing, 2010). NMF became a popular acoustic modeling technique in the late 1990s (Lee, Seung, 1999) and aims to represent a single matrix as a product of two separate matrices containing all positive elements (Müller, 2015). NMF is applicable to a variety of tasks but in audio modelling, it operates on the spectral magnitude of a given signal and decomposes the spectrogram into elementary sound units (Makino, 2018). This approach is most competitive when training data is limited and noise signals stationary (Makino, 2018), however when large amounts of data is available, neural network based methods provide optimal performance. While statistical methods have proven some utility in removing non-stationary noise (Raj, et al., 2010), deep learning methods outperform their statistical counterparts as the complexity of the noise increases (Li, et al., 2015).

For the task of noise removal, DNNs learn a non-linear mapping from observed noisy speech to the desired clean speech signal, enabling reconstruction of the target features (Zhang, et al., 2018). Regardless of the compensation method, noise reduction commonly induces artifacts into the cleaned signal (Huang, Zhang, Zhang, Zou, Zeng, 2014). While including artifacts during ASR network training is a common practice, systems have also been successfully deployed with noise suppression preprocessing and no artifact compensation (Marti, Cobos, Lopez, 2012). Useful DNN architectures for this approach include stacked auto-encoders, LSTM-RNNs, CNNs, and Generative Adversarial Networks (GANs) (Zhang, et al., 2018).

The context-dependent encoding of LSTM-RNN systems is a hallmark of its ability to learn complex non-stationary noise functions (Weninger, Eyben, Schuller, 2014). Deep mask-based approaches have proven useful in removing both additive and convolutional noise from speech (Zhang, et al., 2018). In this method, the network learns an estimation of the desired spectral content, or a mask, and filters noise by applying the mask to the spectrum of the mixed noisy signal (Zhang, et al., 2018). In contrast, GANs utilize two neural networks in sequence. The first maps noisy speech to clean speech, and the second aims to distinguish whether its input comes from enhanced speech or clean speech (Zhang, et al., 2018).

Recently, a system utilizing a CNN-LSTM architecture has advanced the ASR task in multi-talker environments (Wang, Muckenhirn, Wilson, Sridhar, Wu, Hershey, …, Moreno, 2018). The system consists of two individual networks, the first utilizes a speaker identification model and encodes characteristics of the target speaker. The second network uses spectrogram mask-based learning and filtering to remove unwanted multi-talker noise from the target signal (Wang, et al., 2018). This model successfully improves the word error rate of the Neural Speech Recognizer network in multi-talker ASR, while having a minimally adverse affect in the single talker scenario (Wang, et al., 2018). This model has yet to be tested on speech corrupted with environmental noise (Wang, et al., 2018).

In lieu of preprocessing incoming noisy speech data, model-based approaches to noise robust ASR have also proven effective (Zhang, et al., 2018). In model-based compensation, the noisy input signal remains unprocessed and the network learns the relationship between noisy

speech and transcription targets (Zhang, et al., 2018). It can also be proven that noisy training in general can act as a regularizer of training data, thus reducing the model's variance and inclination to overfit the training data (Yin, Liu, Zhang, Lin, Wang, Tejedor, Li, 2015).

While noisy model training coupled with MFCC extraction has proven beneficial (Seltzer, Yu, Wang, 2013), the general trend in ASR is to remove the preprocessing step altogether (Zhang, et al., 2018). End-to-end neural networks are best suited for this approach, as is exemplified in the Deep Speech model, where the input features are simply log-scaled audio power spectra and subsets of the training data were deliberately noisy (Hannun, et al., 2014). Joint approaches consisting of both noise removal preprocessing in addition to model compensation have also shown competitive results (Zhang et al., 2018).

Few standard datasets exist for evaluating noisy speech performance (Hannun, et al., 2014). Perhaps the most prominent is the Aurora series, where each dataset consists of speech corrupted with different types of noise (Li, et al., 2015). These datasets are commonly used in the CHiME speech separation and recognition challenge, wherein participants aim to improve the state of the art of noisy ASR performance (Barker, Marxer, Vincent, Wantanabe, 2015). Common clean speech datasets include the Wall Street Journal, Switchboard, and Fisher corpora (Hannun, et al., 2014). These datasets largely consist of conversational and read speech samples. The general approach to testing performance of ASR in noisy environments involves corrupting a clean speech dataset with some type of noise, and evaluating the system's performance at varying signal to noise ratios (Barker, et al., 2015).

2.5 Exploiting End-to-End Neural Networks

A goal of end-to-end neural networks applied to speech recognition is to remove much if not all pre and post-processing effort while improving model performance. Such networks require large amounts of training data and embody the premise of data-driven approaches that holistic optimization tends to outperform encoding prior knowledge of the given task (Graves, Jaitly, 2014). Classical deep learning systems find the neural network as only one component in a complex pipeline, where the network classifies individual frames of acoustic data and its output distributions are then reformulated as emission probabilities for an HMM, which interfaces with a trained language model (Graves, Jaitly, 2014). In these systems, the acoustic target classification is disjoint from the Word Error Rate objective function, where improvement in acoustic classification does not necessarily relate to transcription accuracy (Graves, Jaitly, 2014). The end-to-end approach seeks to avoid this inconsistency and remove as much expert prior knowledge from the task as possible.

While, end-to-end networks provide state of the art performance on a variety of tasks, they exhibit potential limitations which can be overcome if carefully considered (Glasmachers, 2017). After the initial design of these systems, full automation takes over the learning process. As a network's complexity grows, so does its susceptibility to breakdown during training (Glasmachers, 2017). A common principle of engineering is dividing a large problem into several smaller and more manageable tasks. End-to-end networks ignore this decomposition and rely upon their structure to find solutions in a holistic manner. Thus, even as purely data-driven methods seek to reduce reliance upon preconceived human knowledge, end-to-end networks

are best utilized when designed for the task at hand (Glasmachers, 2017). In this way, harnessing state of the art machine learning methods, an abundance of training data and computational power, in addition to preconceived knowledge of the task may prove beneficial to automatic speech recognition in noisy environments.

3. Procedure

3.1 Procedure Overview

This research focused on the effect of speech separation on the performance of an end-to-end neural network for automatically transcribing additively corrupted speech. This was addressed by experimenting with existing, open source technologies, libraries, and datasets. A noisy speech dataset was created and then processed with statistical and deep learning-based speech separation algorithms. The separation methods included non-negative matrix factorization (NMF) and a bidirectional recurrent neural network (RNN) with long short term memory (LSTM). Each supervised speech separation algorithm was trained using a subset of the previously curated speech and noise data. Each resulting processed dataset was then evaluated using standard source separation evaluation metrics, and the speech content was transcribed by the Deep Speech ASR system. Deep Speech's word error rate when transcribing the separated vocal signals across each processed dataset was then compared, with accuracy on clean, unprocessed speech data as a base line. All code for this research was written in Python.

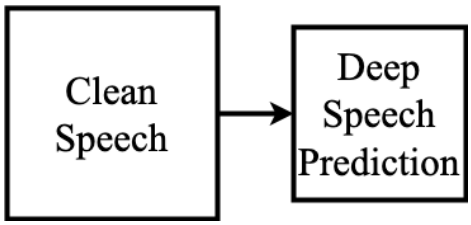

Figure 3.1 Signal flow for determining clean speech baseline word error rate.

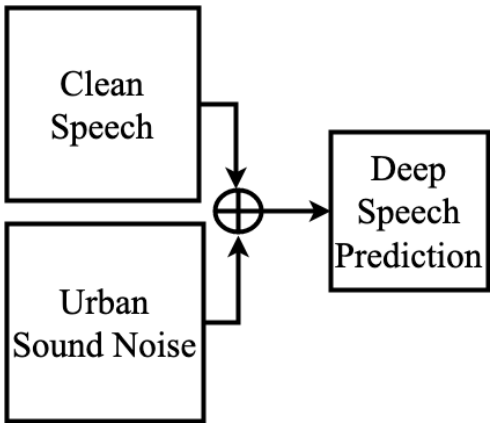

Figure 3.2 Signal flow for determining noisy speech baseline word error rate. Speech and noise samples were normalized to a -30dB reference level and then added together at varying signal to noise ratios of 2, 6, and 10 to 1.

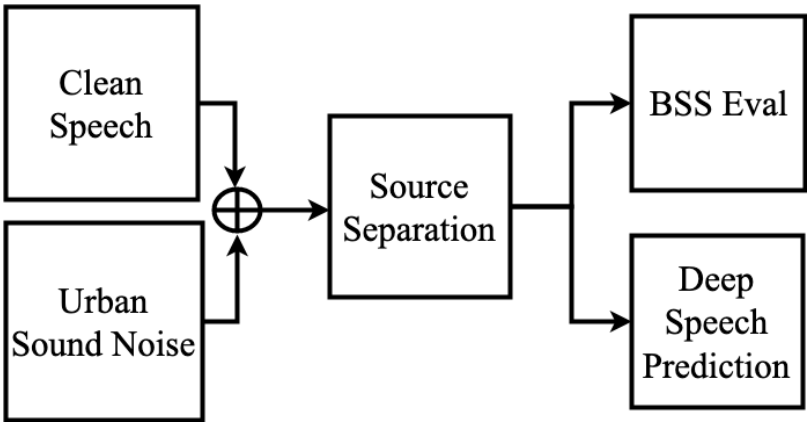

Figure 3.3 Signal flow for determining separated speech word error rate and separation quality.

3.2 Dataset Curation

Training and testing data for this research consisted of two clean speech datasets corrupted with one dataset of real-world urban noise. Multiple noisy speech datasets were created for testing, varying only in their signal to noise ratios. Clean speech was obtained from Mozilla's publicly available Common Voice English dataset as well as the LibriSpeech collection of read speech. Corrupting the clean speech samples was conducted with the Scaper library in Python, which additively combines selected audio sources with adjustable parameters (Salamon, MacConnell, Cartwright, Li, Bello, 2017).

Each speech and noise dataset was loaded and processed according to comma-separated value (CSV) files, providing a consistent metadata footprint throughout the pipeline. When selecting training, validation and testing subsets, the information for each audio file was loaded into data frames using the Pandas library (McKinney, 2011) and then subsequently converted to NumPy arrays (Van Der Walt, Colbert, Varoquaux, 2011). Splitting of training, validation, and testing data, was conducted by generating sequences of random numbers corresponding to the original metadata array indices. The audio file information at these random indices was then copied into new arrays, representing the desired splits of each dataset. Metadata for each random dataset split was then saved to a new CSV file. Relevant processing information at each stage of the pipeline was appended to the input CSV file and subsequently stored. Each dataset was split into 7,000 files for training, 800 for validation, and 800 for testing.

3.2.1 Common Voice

Common Voice is a crowd sourced dataset, containing MP3 recordings with corresponding transcripts uploaded by individuals around the world. At the time of writing, Common Voice contained 780 hours of data from roughly 39,500 speakers, with both male and female voices and a wide range of accents and recording quality. The dataset is continuously growing as individual contributions persist. Users also contribute to the validation of usable data and reliable transcript labelling. A subset of user-validated data was used for this experiment, consisting of 65,500 samples. As a method of objectively controlling sample quality and removing muffled speech, examples with the lowest frequency content above 7 kilohertz were discarded. This was determined by applying a 17th-order, type 2 Chebyshev high-pass filter with a cut-off frequency of 7 kilohertz. Resulting samples with the highest measured power level were then kept for selection of training, validation and test sets. The filtering process was conducted with the SciPy Signal Python library (Virtanen, Gommers, Oliphant, Haberland, Reddy, Cournapeau, ... Van Der Walt (2019). 12,500 examples with the least high frequency content were discarded, leaving 53,000 speech examples for further sampling.

Ultimately, subset of 8,600 files were randomly sampled from the remaining 53,000 samples of the Common Voice dataset. 7,000 of these samples were randomly segmented into a training set, and 800 into a development set to be used for training the source separation algorithms. The remaining 800 samples were used for testing source separation and speech recognition performance. These files were converted from their original MP3 format to 16 bit 44,1000 Hertz WAV files. The labels of the dataset were edited to remove all capitalization and

punctuation except for apostrophes in order to correspond with Deep Speech's classification outputs. Minor spelling errors were corrected and digital numbers were converted to text as well. The training split for Common Voice consisted of 10 hours of speech data, while both validation and test sets contained approximately 70 minutes of audio data each.

3.2.2 LibriSpeech

The LibriSpeech dataset was used in this research in order to test DeepSpeech's performance on speech examples comparable to a subset of its original training data and with inherent quality control. While Common Voice represents a distribution of English speech samples that may be encountered literally across the world, LibriSpeech is more refined and intentionally designed for the task of ASR (Panayotov, Chen, Povey, Khudanpur, 2015). LibriSpeech consists of English speech samples derived from audiobooks read aloud (Panayotov, et al., 2015). This style of speech corresponds to more than half of the data used to train DeepSpeech (Hannun, et al., 2014). The creators of Deep Speech developed their own read speech dataset, contributing the majority of training examples to the network (Hannun, et al., 2014). The Wall Street Journal read speech dataset also contributed to DeepSpeech's training data, and was used as a measure of quality in LibriSpeech's curation (Panayotov, et al., 2015). Acoustic models trained on LibriSpeech performed better on the Wall Street Journal test set than models trained on the Wall Street Journal set itself (Panayotov, et al., 2015).

The creators of LibriSpeech normalized the ground truth labels of the dataset by converting characters to upper-case, removing punctuation and expanding abbreviations (Panayotov, et al., 2015). For this research, additional normalization was conducted to align with Deep Speech's predictions in correspondence with the aforementioned adjustments to Common Voice. The LibriSpeech audio was split into segments of 35 seconds or less, balanced in terms of gender, and recordings were curated to ensure minimal audio quality problems (Panayotov, et al., 2015). The development subset of LibriSpeech was used for the present research, samples of which were deliberately selected as the most challenging of its subsets (Panayotov, et al., 2015). The development set of LibriSpeech consists of 5.4 hours of read speech from 20 female and 20 male individual speakers, with approximately 8 minutes per speaker (Panayotov, et al., 2015). 800 files were randomly sampled from the entire development subset for evaluation, corresponding to the number of test samples used from Common Voice and Urban Sound. The final test set consisted of 100 minutes of audio data. These LibriSpeech samples were not used for training source separation algorithms. This in turn provided information about the generalization of the speech separation algorithms, which were trained solely using Common Voice data.

3.2.3 Urban Sound

The speech data obtained from Common Voice and LibriSpeech was additively corrupted with samples from the Urban Sound dataset. Urban Sound contains 27 hours of annotated, real-world urban noise, balanced across 10 sound classes modelled mostly after New York City noise pollution complaints (Salamon, Jacoby, Bello, 2014). The sound classes include: air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shot, jackhammer, siren, and street music. Each sample is no longer than 4 seconds long and care was taken to ensure that recording quality was useful for training scalable algorithms (Salamon, et al., 2014). The samples were originally collected from a publicly available crowd-sourced database, all encoded with varying sampling rates. For this research, all files were converted to WAV format, with a sample rate of 44,100 Hertz and bit depth of 16. The dataset was originally catered toward addressing sound event detection tasks, and separated into 10 specific cross validation folders in order to provide consistency across studies (Salamon, et al., 2014). In this research, 7,000 samples in the first 8 folders were used for training, 800 samples in the 9th for validation, and 800 from the 10th were used for testing. This resulted in roughly an 80%, 10%, 10% split of the data. 7 hours of noise was used to train the source separation algorithms, with roughly 50 minutes used for validation and testing.

Figure 3.2 Duration of data used for training, validation, and testing.

3.4 Creating Noisy Speech Data

The Scaper Python library for soundscape synthesis was used to create the testing datasets for this research. Scaper provides a highly controllable environment for this kind of dataset creation (Salamon, et al., 2017). Test data was created by additively combining the speech and noise datasets, with consistency across varying signal to noise ratios. Scaper's signal to noise ratio is determined by Loudness Units relative to Full Scale (LUFS), which measures

perceived loudness in common broadcasting mediums (Salamon, et al., 2017). The signal to noise ratios used were 2, 6, and 10 to 1, with an initial reference level of -30 decibels. These are all controllable parameters provided by Scaper. The noisy speech data was created by a one to one mapping of the speech and noise test samples. The one-to-one mapping was ensured by utilizing the CSV files created during sample selection at the onset of the experiment. Each resulting test example contained exactly one noise event combined with one clean speech sample. The length of each output testing speech sample was dictated by the longer of the two input samples being combined.

3.3 Source Separation Methods

3.3.1 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) was used as a baseline classical method of source separation. When applied to audio, NMF represents a single Short-time Fourier Transform (STFT) matrix as a product of two separate matrices, using sparse a sparse number of components (Müller, 2015). The two resulting matrices approximate the magnitude spectrogram of a signal as a weighted sum of its components. One of the matrices represents estimated spectral patterns, or frequencies, and the other indicates points in time where those patterns are present and their approximate gain values (Müller, 2015). The dot product of these two matrices represent a spectral mask of the estimated signal. After decomposition, the signal can be approximately reconstructed by multiplying the spectral mask with the input signal and then applying the inverse Short Time Fourier Transform (ISTFT) (Müller, 2015). Wiener Filtering is a common method of post processing in the NMF pipeline, where the original signal is filtered with the approximated spectral mask (Müeller, 2015). The Wiener Filter typically applies a soft or binary mask to the input signal. In the soft mask approach, some unwanted signal components may persist through the separation process, but with a lower risk of losing potentially vital information from the original signal. The binary mask filters much more selectively and is typically used when the estimated spectral mask is expected to be relatively accurate.

The Mini-batch Dictionary Learning method from the Scikit-learn decomposition library in Python was utilized for creating non-negative spectral masks of the speech and noise audio

STFT matrices (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, ... Vanderplas, 2011). This method is able to provide both positive and negative approximations, however parameters are configurable to enforce non-negativity (Pedregosa, et al., 2011). NMF training was conducted using all 7,000 samples from the training splits of the Common Voice and Urban Sound datasets. For each source, speech and noise, STFTs of each training example were computed using the Librosa library (McFee, McVicar, Balke, Thomé, Raffel, Lee, ... Moore, 2019). The STFT parameters for each example match those used in Open Unmix, with 4,096 DFT points, a hop length of 1,024 samples, and Hann windowing function. Resulting STFTs were appended to a master list and then padded with zeros using NumPy to ensure that each was of equal dimension in width. Each STFT was then concatenated into a single, tall NumPy matrix, which was ultimately the source for training the NMF models. Two models were ultimately trained: one for speech and one noise.

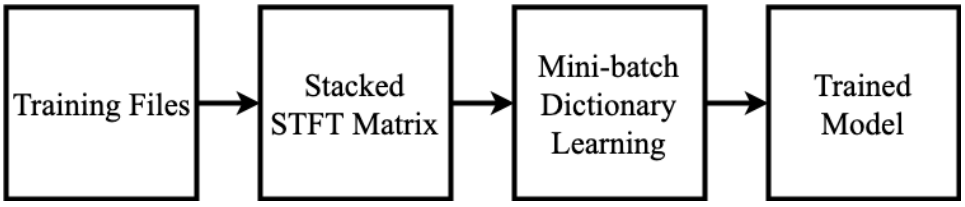

Figure 3.5 NMF modelling signal flow. Training files consist of Common Voice speech and Urban Sound noise for constructing the STFT matrix used to train each respective model.

The number of individual components in which to decompose the signal, as well as the number of optimization iterations are tunable parameters in NMF. The validation subsets of

Common Voice and Urban Sound were used to select parameters for training the NMF model. The final parameters were chosen based upon the optimal Signal to Distortion Ratio provided by the given test parameters when evaluating the development set. For this research, the NMF models used for both speech and noise were trained using 20 components with 1,000 optimization iterations. Parameters considered for each model included all combinations of 15 or 20 components and 500 or 1,000 optimization iterations. Mask estimation was carried out using the decomposition method provided by the Librosa library (McFee, et al., 2019). Testing signals were then reconstructed using the Wiener Filtering approach provided by the Norbert Wiener filter Python library with phase from the original STFT being added to the estimate (Liutkus, Stöter, 2019). Soft mask Wiener Filtering was applied to each of the noisy speech signals to obtain NMF estimates of both speech and noise for each file in the testing set.

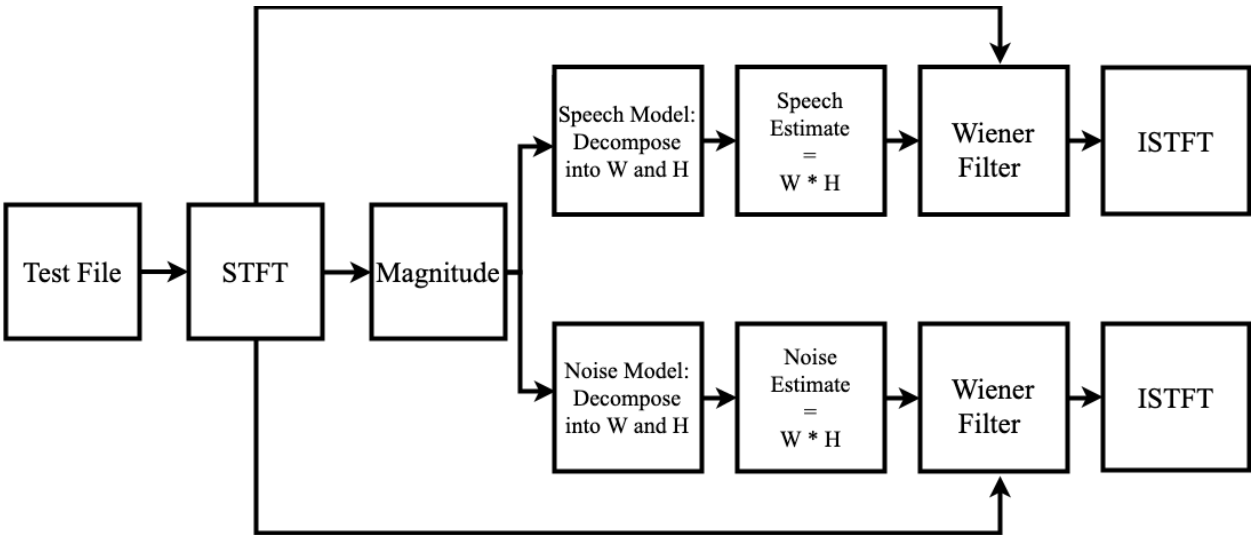

Figure 3.6 NMF processing signal flow. Test files consist of corrupted Common Voice and corrupted LibriSpeech files at 2, 6, and 10 to 1 signal to noise ratios. Final ISTFTs produce the samples evaluated with source separation metrics and Deep Speech.

3.3.2 Open Unmix RNN with LSTM

Open-Unmix is a publicly available neural network intended for source separation, and originally trained for music applications. The project was intended to provide a publicly available state of the art method for flexible audio separation research (Stöter, Uhlich, Liutkus, Mitsufuji, 2019). The original model was trained to separate music signals into 4 components: bass, drums, vocals, and other (Söter, et al., 2019). The network is based in PyTorch, with the goal of allowing researchers to easily extend its application to new representations (Söter, et al., 2019). The Open-Unmix network was chosen for this research because it is considered a state of the art open-source source separation system (Söter, et al., 2019).

At its core, the Open-Unmix model contains three bidirectional LSTM layers and produces a spectral mask of its predicted target (Söter, et al., 2019). The network provides robustness to gain variation by using batch normalization in multiple stages throughout the training process (Söter, et al., 2019). Batch normalization is also known to reduce the training time of such complex networks, while maintaining their representation ability (Ioffe, Szegedy, 2015). Open-Unmix learns target masks by providing magnitude spectrograms of the mixture and target signals during training. Input spectrograms are obtained using 4,096 DFT points with a hop length of 1,024. The model produces spectral mask estimates, with the final separation of target and mixture performed using the Norbert Wiener Filter, which in this research was also used in NMF separation.

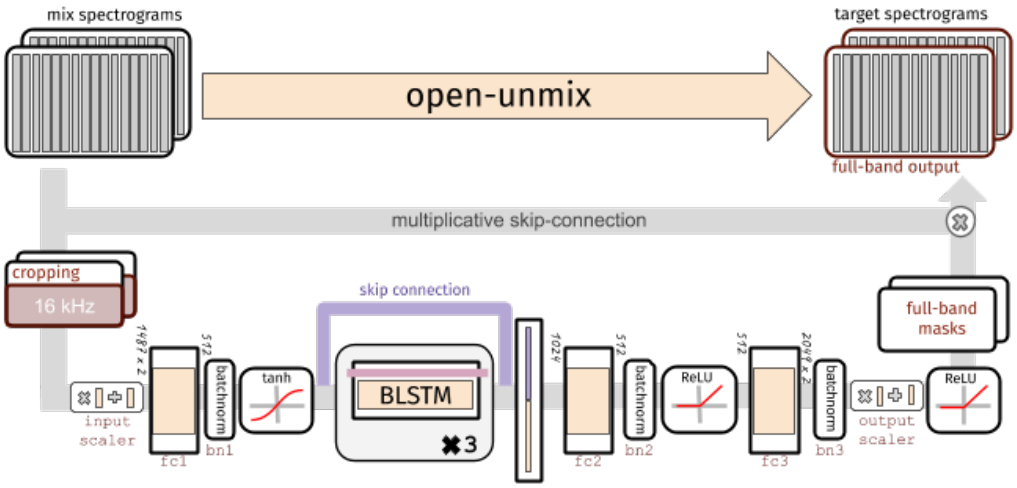

Figure 3.7 Open-Unmix modelling signal flow (Söter, et al., 2019).

In the present research, Open-Unmix was retrained to predict spectral masks for training examples of the Common Voice and Urban Sound datasets separately. Training was conducted using Open-Unmix's command line tools and required the use of New York University's High Performance Computing system for its GPU processing power. The network was trained with 7,000 examples and 800 validation samples, using a batch size of 16, over 1,000 epochs. In this case, validation samples were used for stopping early during training, when the model showed minimal improvement in subsequent epochs. Both bidirectional and unidirectional models were trained in order to evaluate the influence of acoustic context.

Open-Unmix's separation function was run sequentially upon corrupted samples in the testing dataset, processing individual files and obtaining estimates of speech and noise. While Norbert's soft mask filtering was applied for the NMF estimation, a binary mask was utilized during the Unmix separation process. Each speech and noise estimate was saved to a WAV file for later analysis.

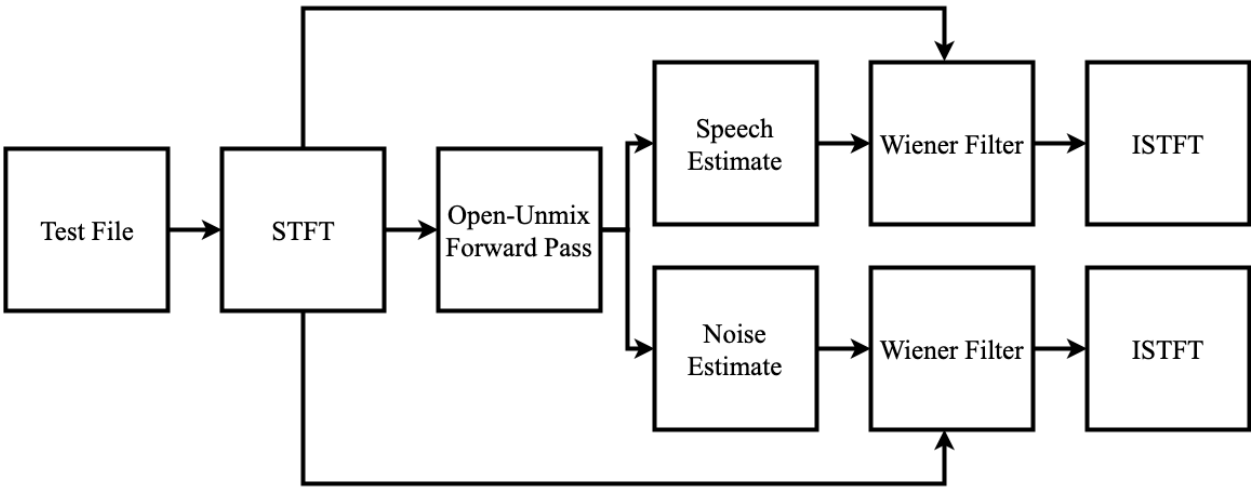

Figure 3.8 Open-Unmix processing signal flow, with the Open-Unmix forward pass being shown in figure 3.7.

3.4 Speech Recognition with Deep Speech

Deep Speech is a publicly available state of the art ASR system, trained end-to-end using synthetically created noisy speech as well as benchmark conversational and read speech corpora (Hannun, et al., 2016). The system ingests sequences of speech spectrograms and converts its input into a sequence of character probabilities, contributing to the final transcription (Hannun, et al., 2016). The characters produced by Deep Speech are letters a through z, apostrophe, and blank, or space (Hannun, et al., 2016). Deep Speech's use of Connectionist Temporal Classification (CTC) eliminates the need for aligned audio and labels during training and testing, removing a significant amount of preprocessing required by classical systems (Graves, Fernández, Gomez, Schmidhuber, 2006).

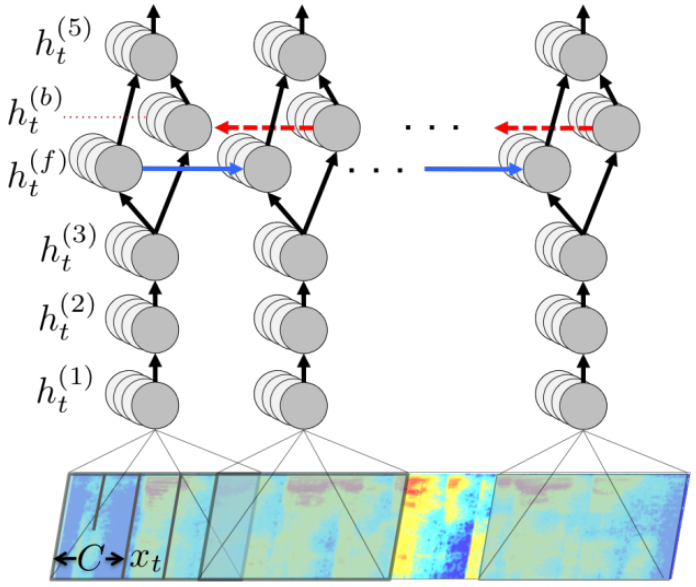

Figure 3.9 Deep Speech layer diagram (Hannun, et al., 2014). This figure displays the 5 layers implemented by Deep Speech, one of which is recurrent and bidirectional.

The creators of Deep Speech put significant effort into curating a large amount of training data (Hannun, et al., 2014). The system was trained on thousands of hours of varied English speech data. Training data included speaker variation, continuous, and noisy speech, intended to provide robust classification in all such circumstances (Hannun, et al., 2014). Deep Speech was trained using over 7,000 hours of speech data from nearly 37,000 speakers, obtained from commonly used benchmark ASR datasets: the Wall Street Journal, Fisher, and Switchboard Corpora (Hannun, et al., 2014). Additionally, noisy speech training data was included by synthetically corrupting speech samples and custom recordings (Hannun, et al., 2014). Synthetic noisy speech training data was intended to contain signal to noise ratios between 2 and 6 to 1, although control over this parameter was not precise (Hannun, et al., 2016).

For this research, all of Deep Speech's default settings were used when producing transcripts from the system's prediction function. The dataset creation pipeline in this research required all testing files to be resampled to 16 bit 44,100 Hz WAV files. As Deep Speech was trained to ingest WAV files sampled at 16,000 Hz, its provided sample rate conversion function was utilized to resample the testing files.

Clean, uncorrupted samples from the Common Voice and LibriSpeech test dataset splits were first processed with Deep Speech in order to obtain baseline transcriptions for further comparison. Transcriptions were then obtained from unprocessed, noisy versions of these samples from each previously applied SNR. Finally, Deep Speech transcriptions were computed when given corrupted speech at each SNR after applying separation with Non-negative Matrix Factorization and Open-Unmix methods. The ensuing prediction, and Word Error Rate for each file were finally stored for later analysis.

3.5 Evaluation

3.5.1 Source Separation Metrics

Standard metrics for blind source separation (BSS) were used for measuring the quality of each separation method. These metrics include signal to distortion ratio (SDR), signal to interference ratio (SIR) and signal to artifact ratio (SAR) (Vincent, Gribonval, Févotte, 2006). These measurements were inspired by the common definition of signal to noise ratio and allow for the independent measurement of interference from unwanted sources (Vincent, et al., 2006). Interference is considered the noise which the separator sought to remove. Artifacts are defined as unwanted noise induced by the source separation algorithm itself, and distortion is akin to the combination of interference and artifacts (Vincent, et al., 2006). SDR is considered the global separation performance measure in the case that interference and artifacts are equally important. Measured in decibels, these metrics are based on the true clean signal, the estimated clean signal, the true noise signal, and the estimated noise signal. For all of these metrics, a higher number indicates better quality, as a large amount of signal energy relative to that of distortion, artifacts, or interference is desired.

The mir_eval Python library was used to measure the BSS metrics for the test data (Raffel, McFee, Humphrey, Salamon, Nieto, Liang, Ellis, 2014). Mir_eval is an open source library that provides a standardized method for evaluating the performance of music and audio information retrieval algorithms (Raffel, et al., 2014). The separation bss_eval_scores method was used to obtain metrics for each separated test signal. The final SDR, SIR, and SAR values, transcript labels, and noise types for each test file were then compiled into a single CSV file.

3.5.2 Speech Recognition Metric: Word Error Rate

Word Error Rate (WER) was used to measure DeepSpeech's performance on each provided dataset. WER is a measure of the edit distance between two strings of text and quantifies the number of operations required to transform the predicted string into its associated ground truth (Wagner, Fischer, 1974). The operations used in transformation include exchanging one word for another, and inserting or deleting a word (Wagner, Fischer, 1974). The algorithm used in this research was supplied by the jiwer python library, and implements the procedure introduced by Wagner and Fischer in 1974 (Vaessen, 2019).

$$\mathbf{WER} = \frac{\mathbf{S} + \mathbf{D} + \mathbf{I}}{\mathbf{N}}$$

S = number of substitutions
D = number of deletions
I = number of insertions
N = number of words in the reference

Figure 3.10 Word error rate equation.

4. Results and Discussion

4.1 Source Separation Results

Source separation metrics were recorded for each source separation method's performance in separating speech data from the Common Voice and LibriSpeech datasets after being corrupted with Urban Sound noise data at controlled signal to noise ratios. The same noise signals used to corrupt the Common Voice test dataset were also used for LibriSpeech. Each respective dataset was exactly the same, varying only in signal to noise ratios of 2, 6, and 10 to 1.

Figures 4.1 and 4.2 provide several points of comparison, rooted in the overall performance measure of signal to distortion ratio. Signal to interference and signal to artifact ratios are reported in Appendix A and exhibit trends similar to the figures below. Additionally, the results of processing data with bidirectional and unidirectional Open-Unmix models were identical and plots referring to Open-Unmix indicate both models.

The immediate point of comparison in the figures below is between each source separation method's overall performance on data corrupted with varying SNRs. For both the Common Voice and LibriSpeech datasets, the Open-Unmix (UMX) model outperforms the Non-negative Matrix Factorization (NMF) model across all SNRs. This is consistent with results found in the literature review, where deep learning models, especially those with memory constructs, tend to outperform their statistically-rooted classical counterparts. Further, the performance of both models on both datasets steadily increases as signal to noise ratio increases, indicating that both models leave room for improvement in terms of SNR robustness.

Both bidirectional and unidirectional UMX models produced identical BSS evaluation results. This suggests that encoding of future temporal context in addition to that of past samples may be most beneficial when processing sounds with a longer duration than those used here. Each noise sample of the Urban Sound dataset is a maximum of 4 seconds in duration and this brevity may have played a role in the minimal impact of additional context encoding.

Although, NMF is unlikely to outperform deep learning models for speech separation from non-stationary noise, a more generalizable NMF model than that used in this research may be obtainable. The NMF model used here contained 20 components and training a model with a higher number of components may improve performance. Additionally, the performance of both NMF and UMX may improve if trained using a dataset curated with more quality control. For example, the LibriSpeech may have provided higher quality speech examples to each model as recording and speech quality was taken into account during its construction. One benefit of the Common Voice dataset is that it provides a rich diversity of examples collected from all over the world. However, this crowd-sourced approach allows for an inherent amount of noise in the data.

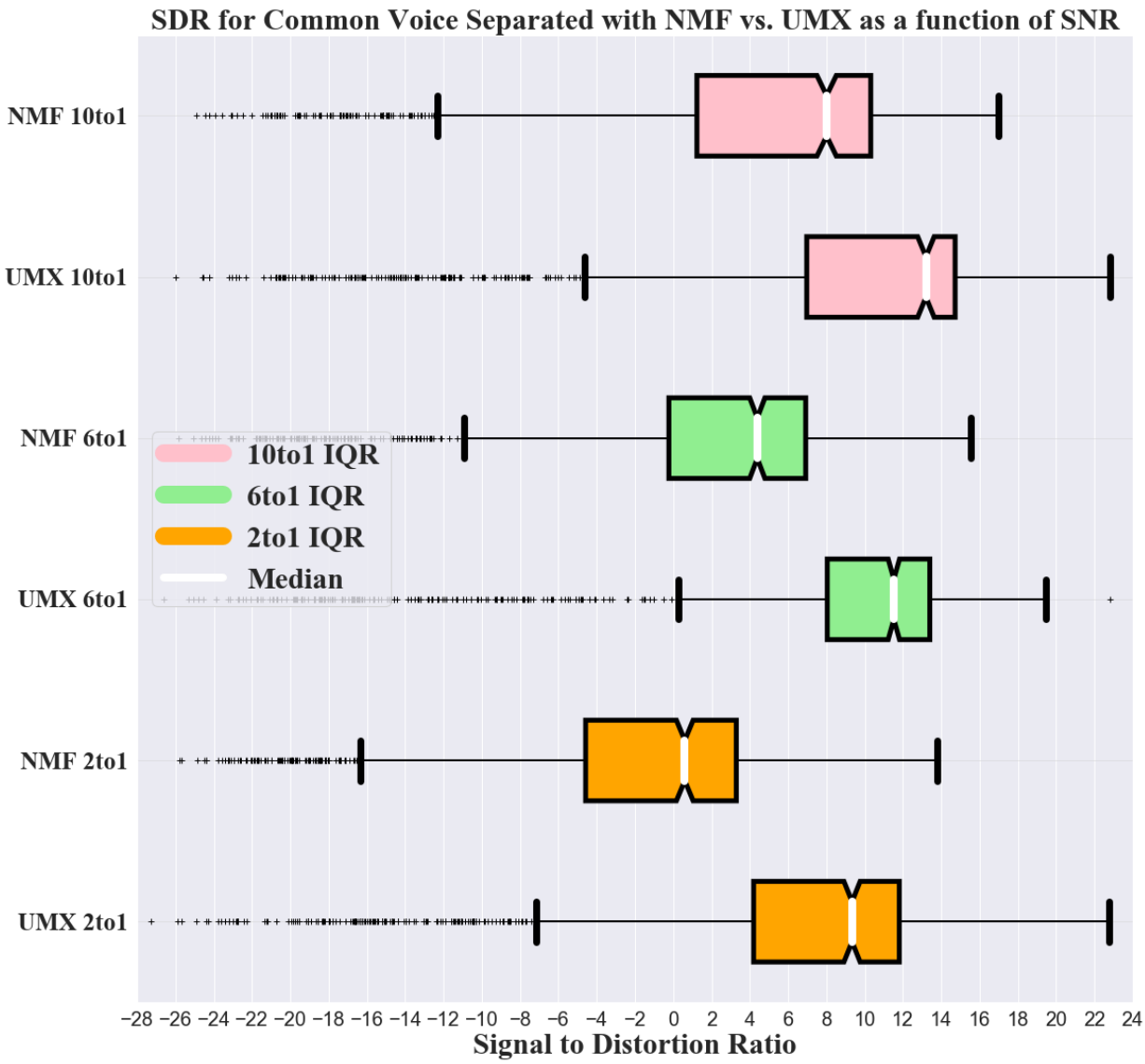

Figure 4.1 Common Voice signal to distortion ratio (SDR) distribution.
SDR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at signal to noise ratios of 2, 6, and 10 to 1.

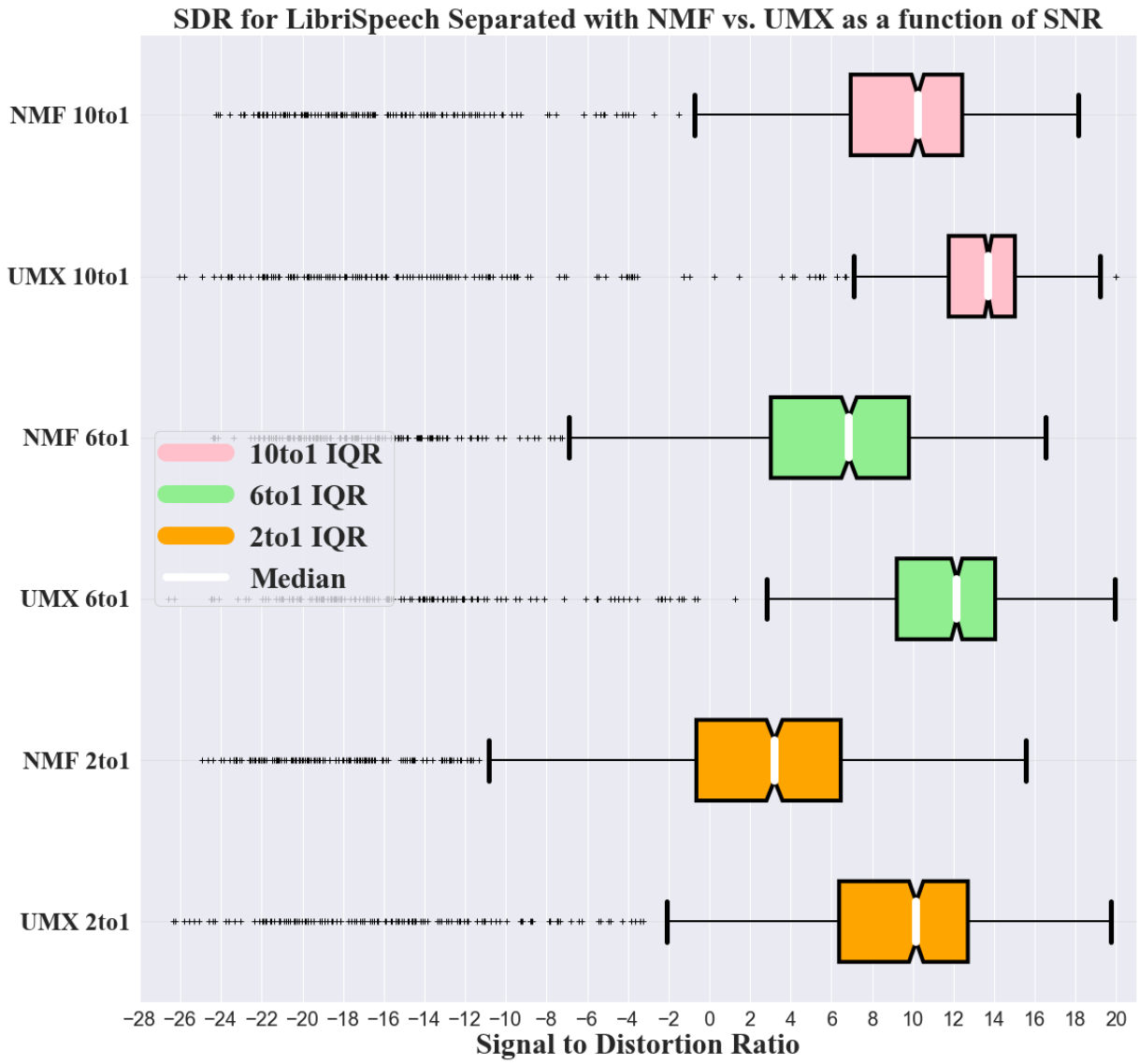

Figure 4.2 LibriSpeech signal to distortion ratio (SDR) distribution.
SDR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at signal to noise ratios of 2, 6, and 10 to 1.

In addition to the figures above, Signal to Interference Ratios (SIR) and Signal to Artifact Ratios (SAR) reported in Appendix A also provide insightful information. UMX outperforms NMF in terms of SIR, indicating that it more consistently removes noise from the targeted speech signal. However, SAR values tend to be higher for NMF than UMX. This speaks to the overall effect that NMF has on the noisy speech signal, which appears to be minimal in terms of removing noise, as well as imposing any spectral mask. UMX consistently removes noise at each SNR tested, while also consistently inducing separation artifacts. NMF is less effective at removing the noise but generally does not appear to degrade the signal any further throughout its processing.

Each model's performance on specific types of noise provide information about which types of complex nonlinear functions they are able to filter. Figures 4.3 and 4.4 display the SDR performance of NMF and UMX when evaluating speech samples corrupted with each of the 10 noise types present in the Urban Sound dataset at a fixed SNR of 6 to 1. The range of noise types used in this research represent both transient and continuously evolving sounds. The gunshot noise type is the most latent of the transient sounds, however many car horn and dog bark examples in the dataset are also relatively transient in nature. The UMX median performance across all noise types does not vary widely relatively, indicating its ability to model stationary as well as non-stationary noise signals. Its lowest performing noise types include dog barking and children playing, both of which exhibit spoken speech characteristics, which may partially explain the performance results. NMF on the other hand consistently provides

higher a higher SDR when separating speech from transient signals such as gunshots. This is consistent with expectations, as there is no memory encoded in the NMF model, limiting its encoding of non-stationary signals.

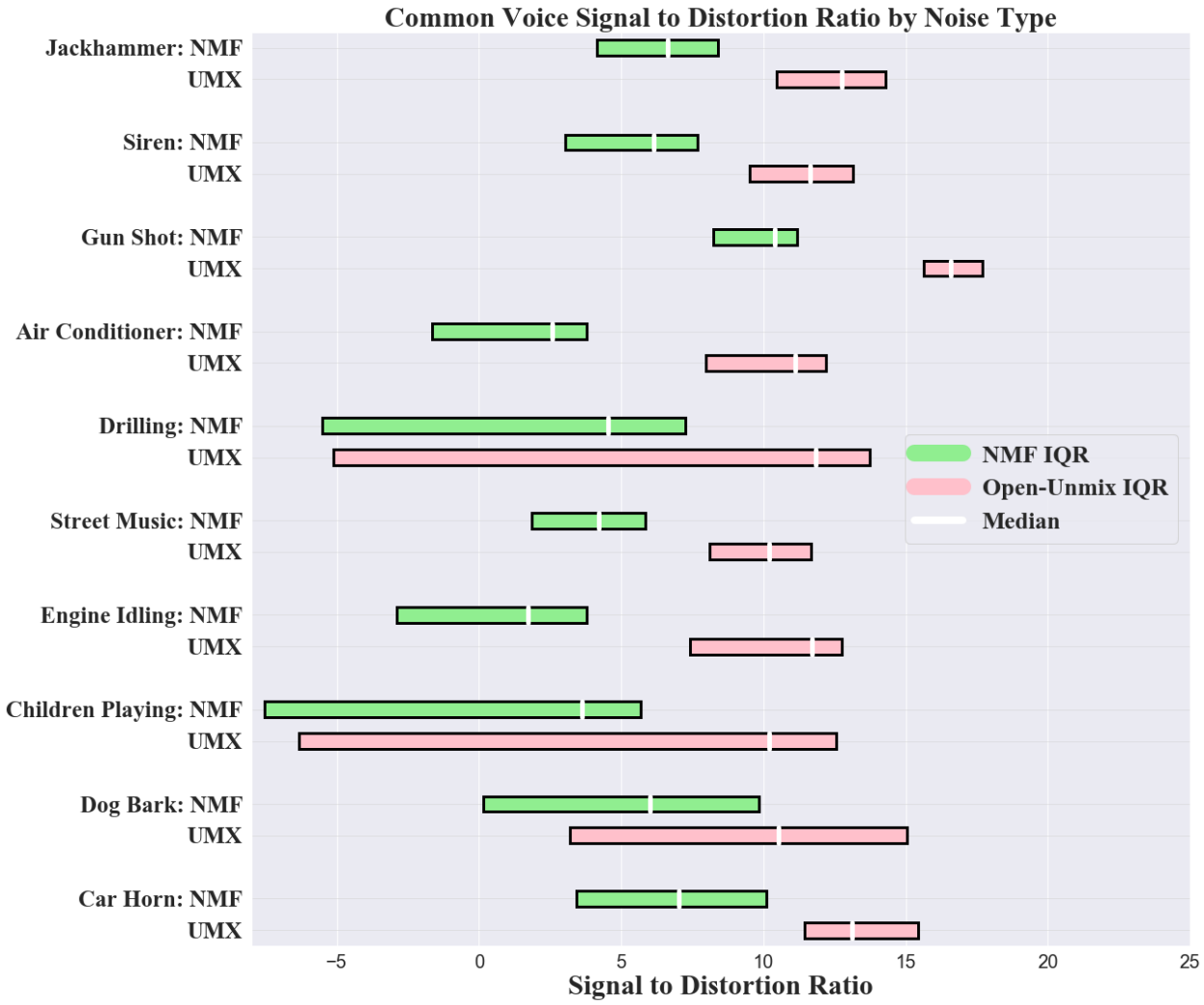

Figure 4.3 Common Voice signal to distortion ratio (SDR) distribution as a function of noise type. SDR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at a fixed signal to noise ratio of 6 to 1. Varying noise and separation types are shown on the y-axis.

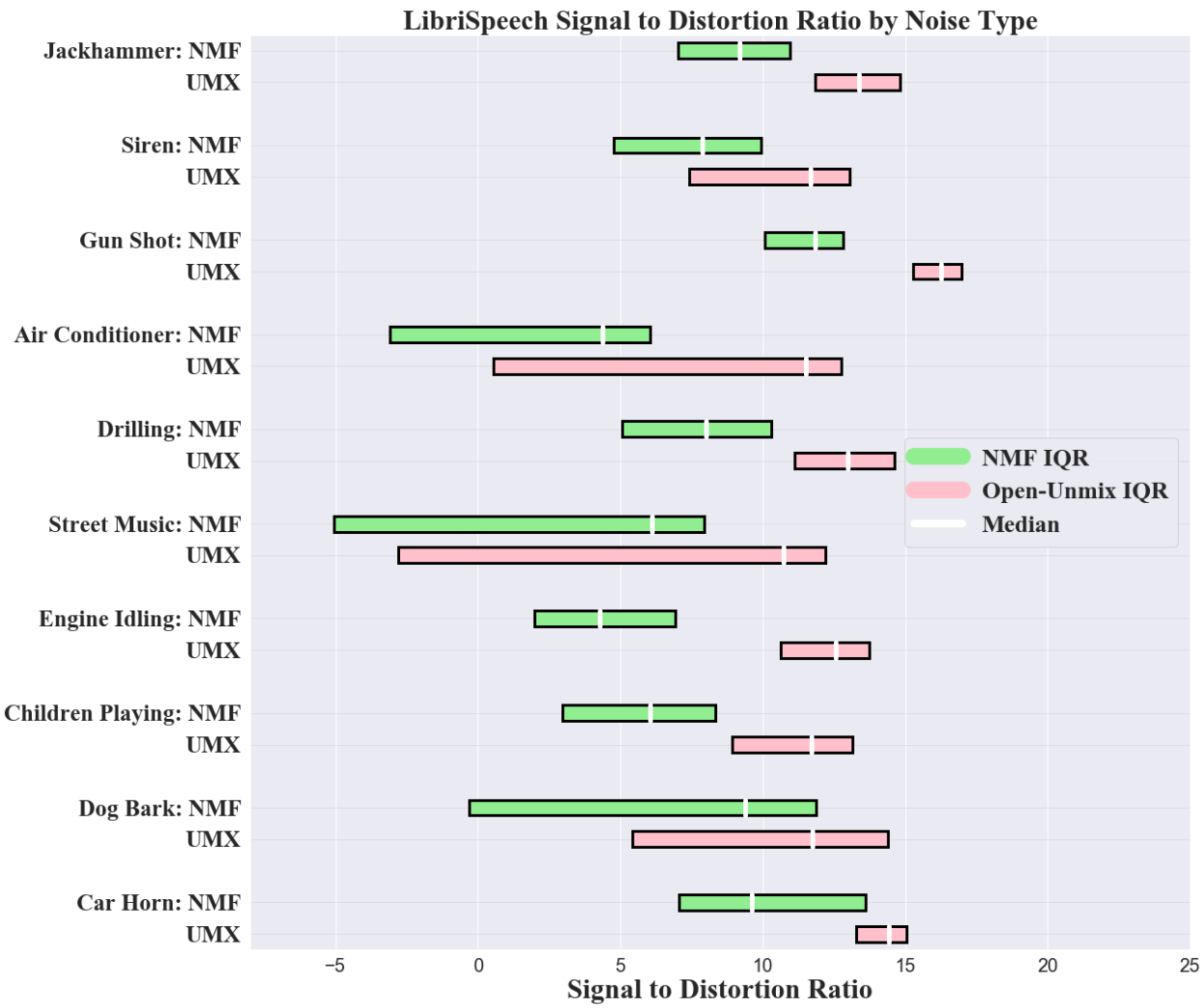

Figure 4.4 LibriSpeech signal to distortion ratio (SDR) distribution as a function of noise type. SDR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at a fixed signal to noise ratio of 6 to 1. Varying noise and separation types are shown on the y-axis.

4.2 Deep Speech Results

Deep Speech's Word Error Rate was measured when given raw speech samples obtained from the Common Voice and LibriSpeech datasets, synthetically corrupted speech, and corrupted speech after processing with Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX). Results were recorded for synthetic corruption signal to noise ratios of 2, 6, and 10 to 1. An initial comparison of Deep Speech's WER when given clean speech samples obtained from Common Voice and LibriSpeech show significantly higher accuracy for the LibriSpeech dataset. This suggests that Deep Speech is indeed more accurate when presented with data recorded under quality controlled conditions and speech types similar to a subset of its training data. This also supports the idea that Common Voice data is inherently noisier than LibriSpeech by nature. Deep Speech's performance on clean speech data additionally indicates the model's stability in this research, as results here are comparable to that in the original research (Hannun, et al., 2014).

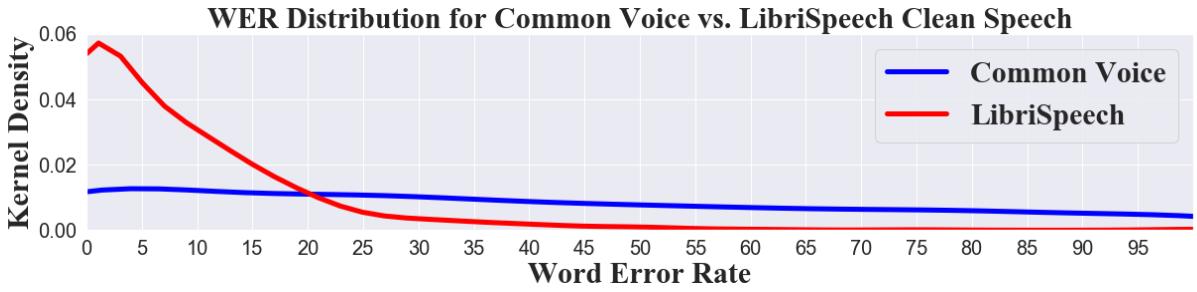

Figure 4.5 Deep Speech word error rate distribution when predicting clean speech.

Figures 4.6 and 4.7 present Deep Speech's WER when presented with clean speech, synthetically corrupted speech at varying signal to noise ratios, and corrupted speech after separation with NMF and UMX. These results demonstrate that the Deep Speech model was less accurate when presented with corrupted speech at all given signal to noise ratios. This initially suggests that there is room for improvement in Deep Speech's modeling of speech corrupted by noise types presented the Urban Sound dataset. Despite a concerted effort for noise robustness in Deep Speech's end-to-end modeling, performance is inhibited by the noise samples presented in this research. Deep Speech's reported WERs on its most challenging test sets were 31.8% and 19.3%, which are comparable to the LibriSpeech dataset corrupted with noise at 10 to 1 SNR and the raw Common Voice dataset (Hannun, et al., 2014). WERs when given clean LibriSpeech data were significantly lower than any other reported dataset. In this research, as SNR decreased so did Deep Speech's accuracy, progressively until results were no longer comparable to the original paper.

Further, Deep Speech's performance is minimally affected by NMF preprocessing and significantly affected by preprocessing with Open-Unmix across both speech datasets. Statistical significance of Open-Unmix's improvement of WER and NMF's lack thereof was determined by one way Analysis of Variance (ANOVA) tests provided by the statsmodels Python API . These results are consistent across all tested SNRs. This supports conclusions found in the literature, which indicate that source separation tends to improve ASR performance. However when applied to Deep Speech, only the RNN-LSTM-based separation provided significant improvement. Thus, the modern, complex and context-encoding acoustic modeling of

Open-Unmix outperformed the classical approach of NMF in this research. Both bidirectional and unidirectional UMX models provided identical WER results, consistent with results reported in the previous section. The figures below indicate both models when referring to UMX. WER results are also consistent with SDR results reported in the previous section; both results sets exhibit trends toward higher performance as SNR increases, where a higher SDR in the previous section is consistent with lower WERs presented here. This trend suggests that as noise decreases, both separation performance and WER improve. Further, for the tests conducted here, as separation performance improves so does WER. Finally, the fact that benefits provided by UMX processing span across both the Common Voice and LibriSpeech datasets speaks to the UMX model's generalizability. Although the UMX speech model was trained solely using Common Voice data, it effectively generalized across the given datasets and improved Deep Speech's performance when analyzing LibriSpeech data.

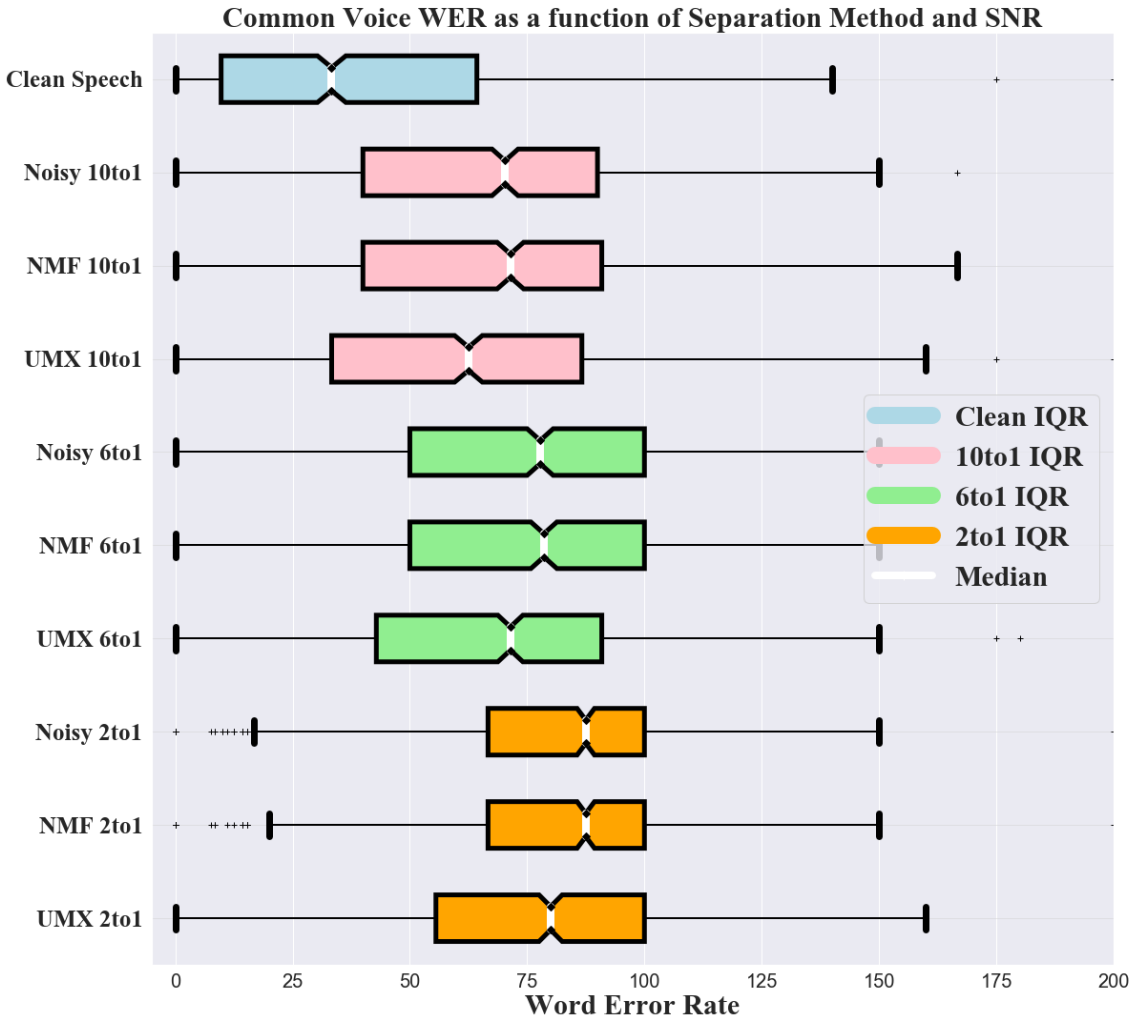

Figure 4.6 Deep Speech word error rate on Common Voice as a function of separation method and signal to noise ratio. The figure displays Deep Speech's word error rate when given raw speech samples obtained from Common Voice (Clean), synthetically corrupted speech (Noisy), and corrupted speech after processing with Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX). Results are shown for synthetic corruption signal to noise ratios of 2, 6, and 10 to 1.

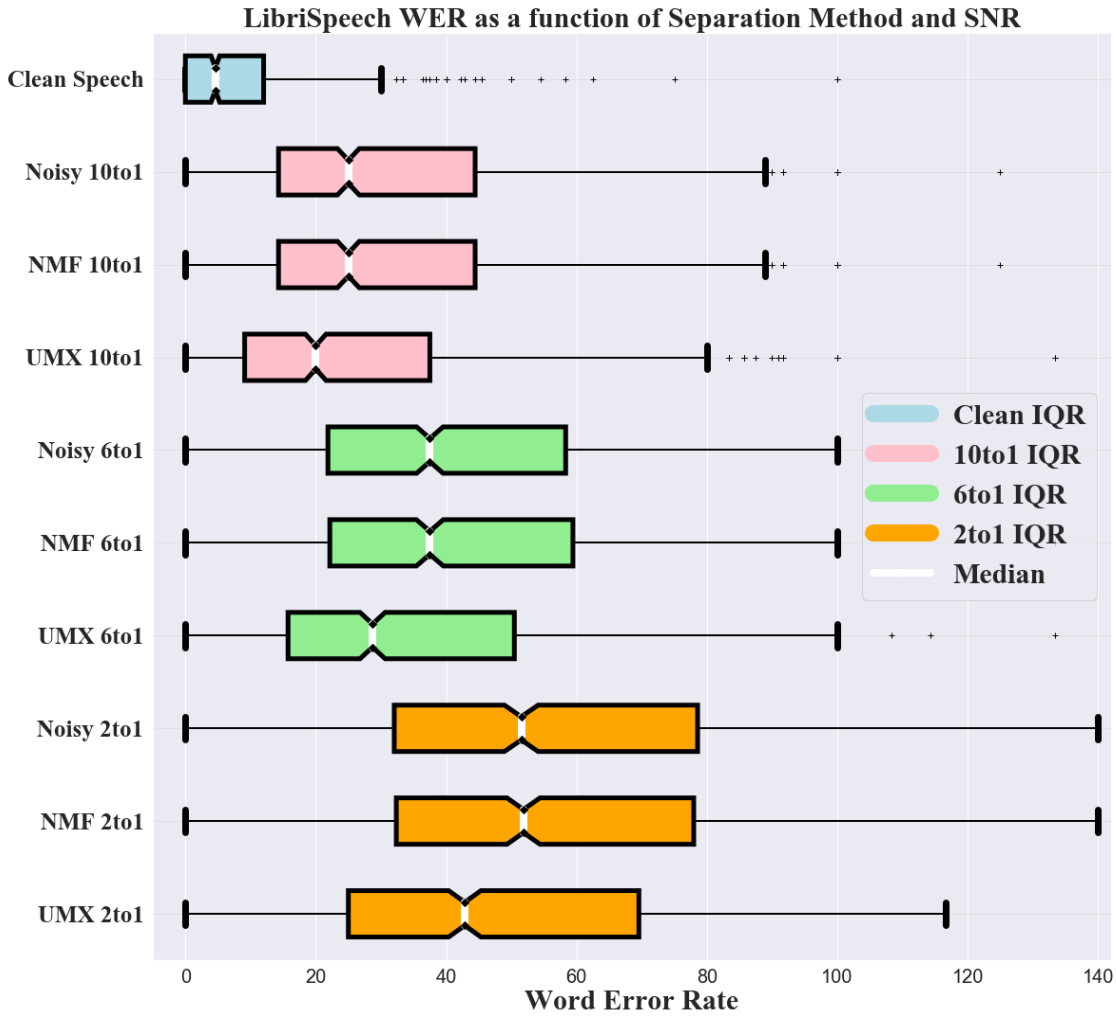

Figure 4.7 Deep Speech word error rate on LibriSpeech as a function of separation method and signal to noise ratio. The figure displays Deep Speech's word error rate when given raw speech samples obtained from LibriSpeech (Clean), synthetically corrupted speech (Noisy), and corrupted speech after processing with Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX). Results are shown for synthetic corruption signal to noise ratios of 2, 6, and 10 to 1.

Measuring Deep Speech's performance as a function of noise type provides insight into its model's robustness to noise with varying sonic characteristics. Figures 4.8 and 4.9 display Deep Speech's WER with and without NMF and UMX preprocessing as a function of noise type at a fixed signal to noise ratio of 6 to 1. These results demonstrate that for both the Common Voice and LibriSpeech datasets, Deep Speech was most robust to transient noise; all gunshot, and many car horn, and dog bark noise examples were relatively transient in nature and WER performance on these noise types is lower than the remaining noise types. For the most part, NMF has a minimal effect on performance in these figures, however it does slightly improve the median WER for the car horn noise type for the Common Voice dataset. This is consistent with NMF's performance on relatively transient data, which necessitates less contextual encoding than non-stationary sounds. UMX consistently improved Deep Speech's median WER on all noise types, further speaking to the generalizability of its modeling.

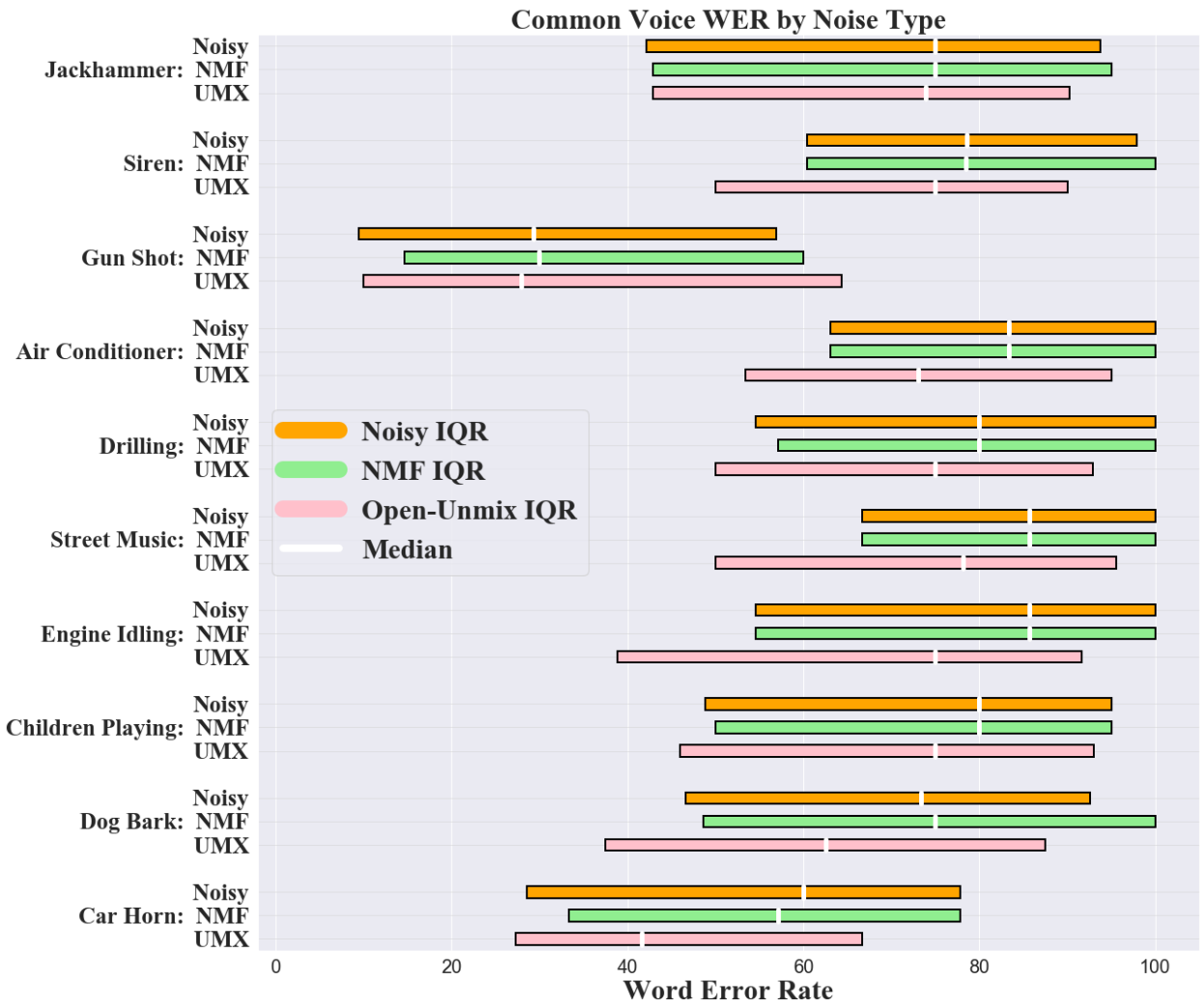

Figure 4.8 Deep Speech word error rate on corrupted Common Voice as a function of noise type. The figure shows the word error rate for corrupted speech unseparated (Noisy), as well as speech processed with Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) with a fixed signal to noise ratio of 6 to 1.

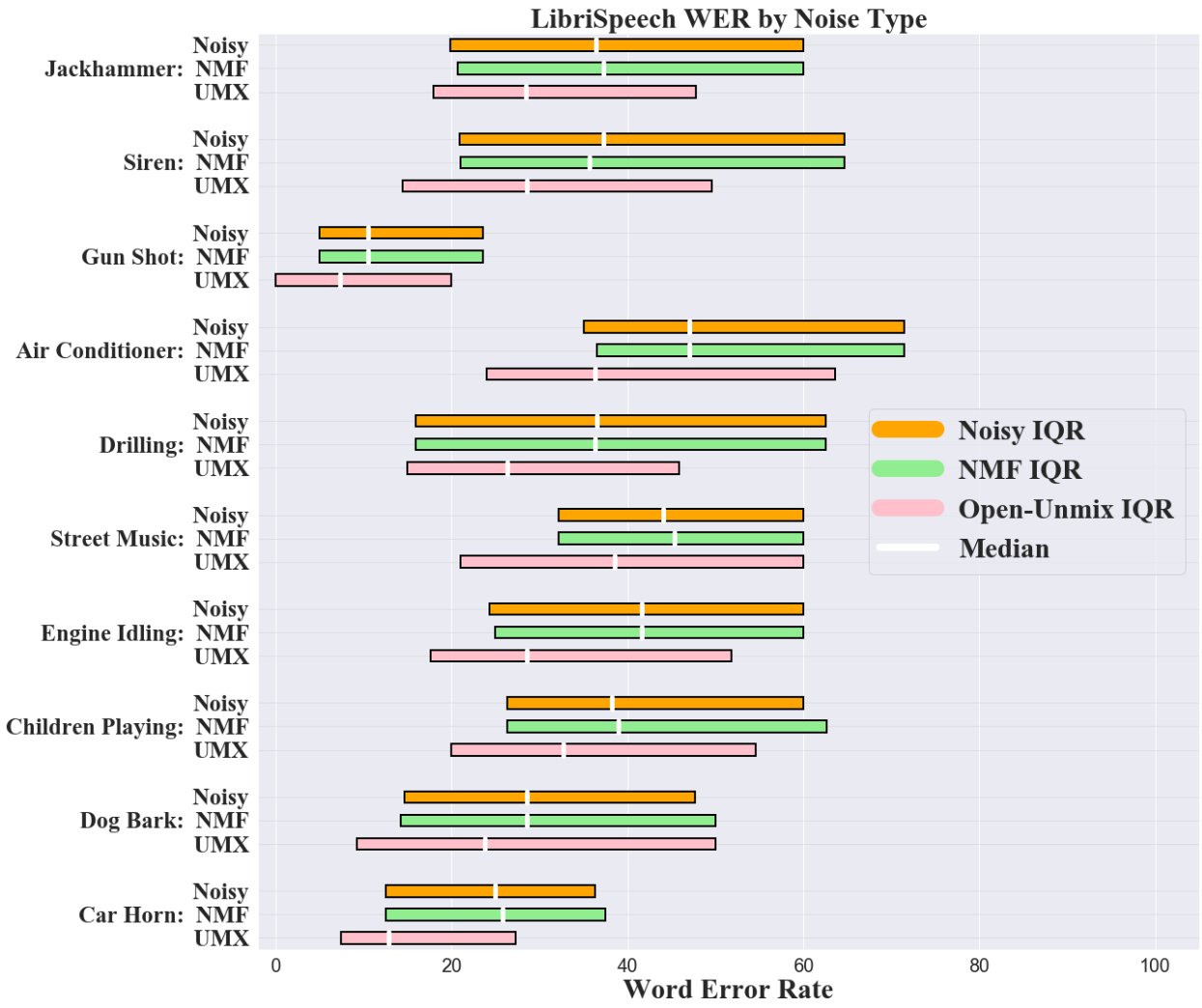

Figure 4.9 Deep Speech word error rate on corrupted LibriSpeech as a function of noise type. The figure shows the word error rate for corrupted speech unseparated (Noisy), as well as speech processed with Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) with a fixed signal to noise ratio of 6 to 1.

4.3 Future Work

This work was constrained to the study of additive noise suppression, and tested only one modern noise suppression method and one end-to-end speech recognition system with select noise classes. Future work should include expanding the scope of noise suppression methods, the types of noise, and the end-to-end speech recognition model itself. Testing additional noise suppression methods under this same process may provide insight into the general applicability of noise suppression to end-to-end speech recognition systems. As U-Net CNN-based separation has proven successful with and without incorporation of memory constructs, these methods should be tested specifically in comparison to the Open-Unmix LSTM model. Experiments using the same process but with convolutive noise would also be a natural progression from this work. Finally, it may be useful to control for the speech recognition system itself. This may come in the form of testing additional ASR systems that were trained end-to-end, training new, or adding noise modelling robustness to existing end-to-end speech recognition systems in light of information provided by this work and future research of this topic.

5. Conclusion

This work studied the effect of noise suppression preprocessing on the accuracy of a speech recognition system that was trained end-to-end. The reliability of classical speech recognition systems depended upon inventive and often arduous preprocessing techniques. However, modern end-to-end systems seek to remove preprocessing from the pipeline, enabling predictions that are based on raw input data. High volumes of training data, consisting of corrupted speech, enable end-to-end neural networks to learn the patterns of the corrupted speech signals. These networks achieve state of the art results for the noisy speech recognition task, however room for improvement remains. This research inquired whether improvement could be found in the form of noise suppression preprocessing and found affirmative results.

Two speech datasets were tested along with one noise dataset. The Common Voice speech dataset was utilized to test wider generalizability, as it consists of crowd-sourced speech samples with little to no quality control. The LibriSpeech dataset was used to measure Deep Speech's accuracy when given speech samples resembling the more stable conditions of its training data; Deep Speech was trained mostly using synthetically corrupted read-speech samples and LibriSpeech samples consist of speakers reading audiobooks aloud and its audio quality is explicitly controlled by its creators. The Urban Sound dataset was used to corrupt subsets of both speech datasets with environmental noise commonly found in cities and at varying signal to noise ratios of 2, 6, and 10 to 1.

The noise suppression methods tested in this work include one classical method, as well as one state-of-the art, modern approach: Non-negative Matrix Factorization (NMF) and a Deep Recurrent Neural Network with Long Short-Term Memory (LSTM-RNN). Both unidirectional and bidirectional LSTM-RNN models were trained and evaluated but exhibited identical results. This may be due to the fact that analysis noise in this research was of relatively short duration, perhaps requiring minimal temporal context in the modeling process. The quality of each separation method was first measured using standard, blind source separation metrics, where the LSTM-RNN approach significantly outperformed NMF processing.

Finally, the Word Error Rate (WER) of a modern speech recognition system trained end-to-end was compared when given corrupted, unprocessed speech data and corrupted speech data after being processed with NMF and the LSTM-RNN. Deep Speech's WER was significantly improved by the LSTM-RNN preprocessing, while NMF provided minimal to no change in accuracy. These results exhibited the same trend across varying signal to noise ratios of 2, 6, and 10 to 1. This indicates that, indeed the accuracy of this modern end-to-end speech recognition system benefitted from neural network-based noise suppression when transcribing speech corrupted with additive environmental noise. This indicates room for improvement in the quality of noise robust speech modeling provided by the Deep Speech ASR system, and affirms the current utility of noise suppression preprocessing for the end-to-end network. Presently, Deep Speech's end-to-end speech transcription benefited from task informed preprocessing.

References

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., ... & Chen, J. (2016, June). Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning* (pp. 173-182).

Barker, J., Marxer, R., Vincent, E., & Watanabe, S. (2015, December). The third 'CHiME'speech separation and recognition challenge: Dataset, task and baselines. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)* (pp. 504-511). IEEE.

Dahl, G. E., Yu, D., Deng, L., & Acero, A. (2011, May). Large vocabulary continuous speech recognition with context-dependent DBN-HMMS. In ICASSP (pp. 4688-4691).

Glasmachers, T. (2017, November). Limits of End-to-End Learning. In *Asian Conference on Machine Learning* (pp. 17-32).

Graves, A., Fernández, S., Gomez, F., & Schmidhuber, J. (2006, June). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd international conference on Machine learning (pp. 369-376). ACM.

Graves, A., & Jaitly, N. (2014, January). Towards end-to-end speech recognition with recurrent neural networks. In International conference on machine learning (pp. 1764-1772).

Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., ... & Ng, A. Y. (2014). DeepSpeech: Scaling up end-to-end speech recognition.

Hannun, A. Y., Maas, A. L., Jurafsky, D., & Ng, A. Y. (2014). First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns. arXiv preprint arXiv: 1408.2873.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. Nature, 401(6755), 788.

Li, J., Deng, L., Haeb-Umbach, R., & Gong, Y. (2015). Robust automatic speech recognition: a bridge to practical applications. Academic Press (pp. 1-60).

Liutkus, A., & Stöter, F.-R. (2019, September). sigsep/norbert: v0.2.1. doi:10.5281/zenodo.3386463.

Makino, S. (Ed.). (2018). Audio Source Separation. Springer (pp. 1-46).

Marti, A., Cobos, M., & Lopez, J. J. (2012). Automatic speech recognition in cocktail-party situations: A specific training for separated speech. *The Journal of the Acoustical Society of America*, *131*(2), 1529-1535.

McFee, B., McVicar, M., Balke, S., Thomé, C., Raffel, C., Lee, D., ... & Moore, J. (2019). librosa/librosa: 0.7. 1. *Zenodo*.

McKinney, W. (2011). pandas: a foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, *14*.

Mozilla (2017). *Mozilla Common Voice.* https://voice.mozilla.org/en

Müller, M. (2015). Fundamentals of music processing: Audio, analysis, algorithms, applications. Springer (pp. 415 - 468).

Panayotov, V., Chen, G., Povey, D., & Khudanpur, S. (2015, April). Librispeech: an ASR corpus based on public domain audio books. In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 5206-5210). IEEE.

Park, T. H. (2009). Introduction to digital signal processing: Computer musically speaking. World Scientific (pp. 333-373).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, *12*(Oct), 2825-2830.

Radha, V., & Vimala, C. (2012). A review on speech recognition challenges and approaches. *doaj. org*, *2*(1), 1-7.

Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., Ellis, D. P. W. (2014). mir_eval: A transparent implementation of common MIR metrics. In *In Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR*.

Raj, B., Virtanen, T., Chaudhuri, S., & Singh, R. (2010). Non-negative matrix factorization based compensation of music for automatic speech recognition. In *Eleventh Annual Conference of the International Speech Communication Association*.

Sak, H., Senior, A., Rao, K., Irsoy, O., Graves, A., Beaufays, F., & Schalkwyk, J. (2015, April). Learning acoustic frame labeling for speech recognition with recurrent neural networks. In 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 4280-4284). IEEE.

Salamon, J., Jacoby, C., & Bello, J. P. (2014, November). A dataset and taxonomy for urban sound research. In Proceedings of the 22nd ACM international conference on Multimedia (pp. 1041-1044). ACM.

Salamon, J., MacConnell, D., Cartwright, M., Li, P., & Bello, J. P. (2017, October). Scaper: A library for soundscape synthesis and augmentation. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)* (pp. 344-348). IEEE.

Seltzer, M. L., Yu, D., & Wang, Y. (2013, May). An investigation of deep neural networks for noise robust speech recognition. In *2013 IEEE international conference on acoustics, speech and signal processing* (pp. 7398-7402). IEEE.

Soltau, H., Liao, H., & Sak, H. (2017). Neural Speech Recognizer: Acoustic-to-Word LSTM Model for Large Vocabulary Speech Recognition. Proc. Interspeech 2017, 3707-3711.

Statsmodels's Documentation. (2009). Retrieved November 2019, from https://www.statsmodels.org/stable/index.html.

Stöter, F.R., Uhlich, S., Liutkus, A., Mitsufuji, Y. (2019). Open-Unmix - A Reference Implementation for Music Source Separation. Journal of Open Source Software, Open Journals, 4(41), 1667.

Van Der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, *13*(2), 22.

Vaessen, N. (2019, February). jiwer. Retrieved October 2019, from https://github.com/jitsi/asr-wer/.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... & van der Walt, S. J. (2019). SciPy 1.0--Fundamental Algorithms for Scientific Computing in Python. *arXiv preprint arXiv:1907.10121*.

Vincent, E., Gribonval, R., & Févotte, C. (2006). Performance measurement in blind audio source separation. IEEE transactions on audio, speech, and language processing, 14(4), 1462-1469.

Wagner, R. A., & Fischer, M. J. (1974). The string-to-string correction problem. Journal of the ACM (JACM), 21(1), 168-173.

Wang, Q., Muckenhirn, H., Wilson, K., Sridhar, P., Wu, Z., Hershey, J., ... & Moreno, I. L. (2018). Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking. arXiv preprint arXiv:1810.04826.

Washani, N., & Sharma, S. (2015). Speech recognition system: A review. *International Journal of Computer Applications*, *115*(18).

Weninger, F., Eyben, F., & Schuller, B. (2014, May). Single-channel speech separation with memory-enhanced recurrent neural networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 3709-3713). IEEE.

Yin, S., Liu, C., Zhang, Z., Lin, Y., Wang, D., Tejedor, J., ... & Li, Y. (2015). Noisy training for deep neural networks in speech recognition. *EURASIP Journal on Audio, Speech, and Music Processing*, *2015*(1), 2.

Zhang, Y., Chan, W., & Jaitly, N. (2017, March). Very deep convolutional networks for end-to-end speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4845-4849). IEEE.

Zhang, Z., Geiger, J., Pohjalainen, J., Mousa, A. E. D., Jin, W., & Schuller, B. (2018). Deep learning for environmentally robust speech recognition: An overview of recent developments. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *9*(5), 49.

Appendix A

Plotting Additional Source Separation Results

This appendix displays the signal to interference and artifact ratios for each source separation method when processing speech corrupted at various signal to noise ratios.

A.1 Signal to Interference Ratios

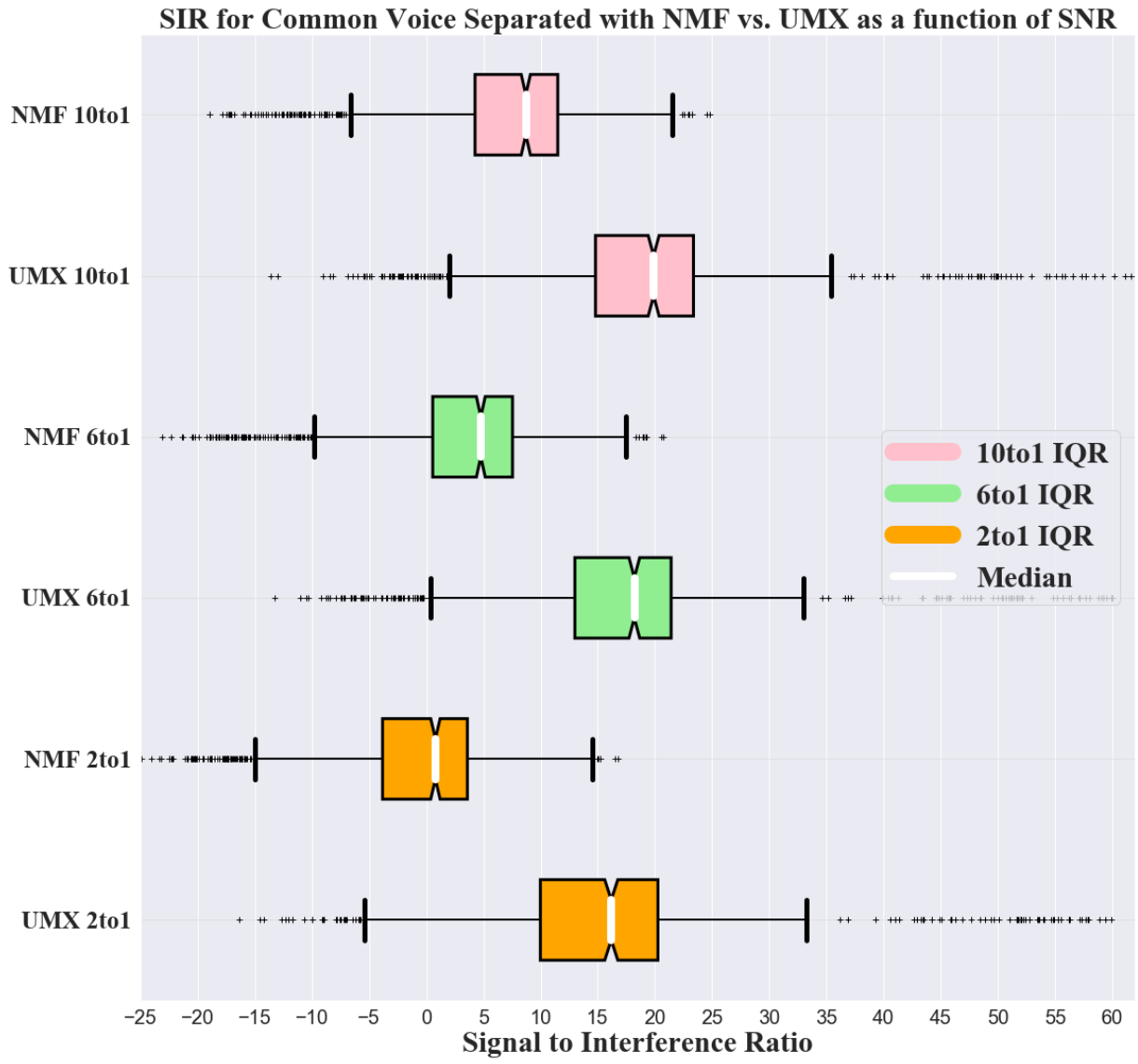

Figure A.1 Common Voice signal to interference ratio (SIR) distribution.
SIR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at signal to noise ratios of 2, 6, and 10 to 1.

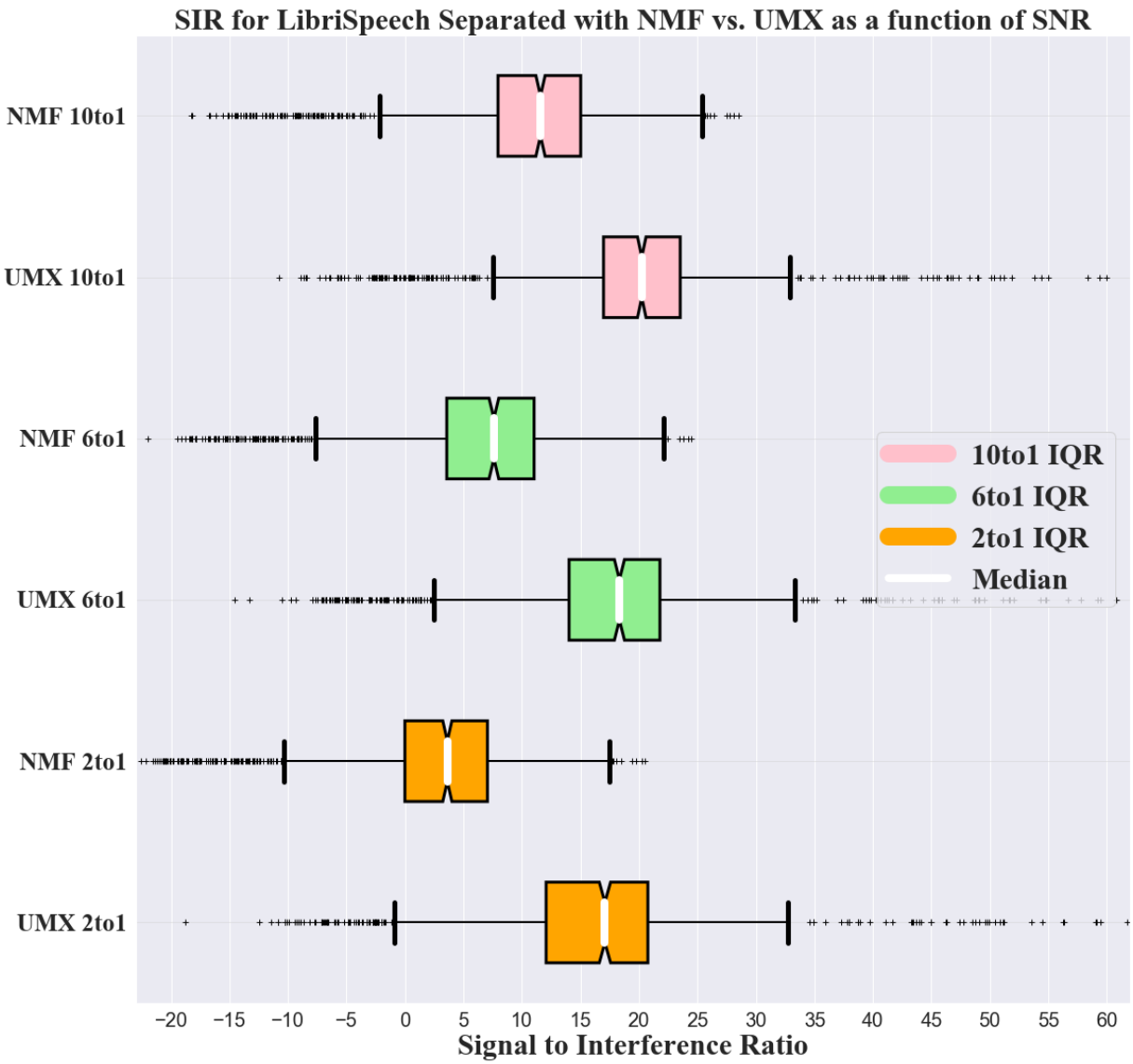

Figure A.2 LibriSpeech signal to interference ratio (SIR) distribution.
SIR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at signal to noise ratios of 2, 6, and 10 to 1.

A.2 Signal to Artifact Ratios

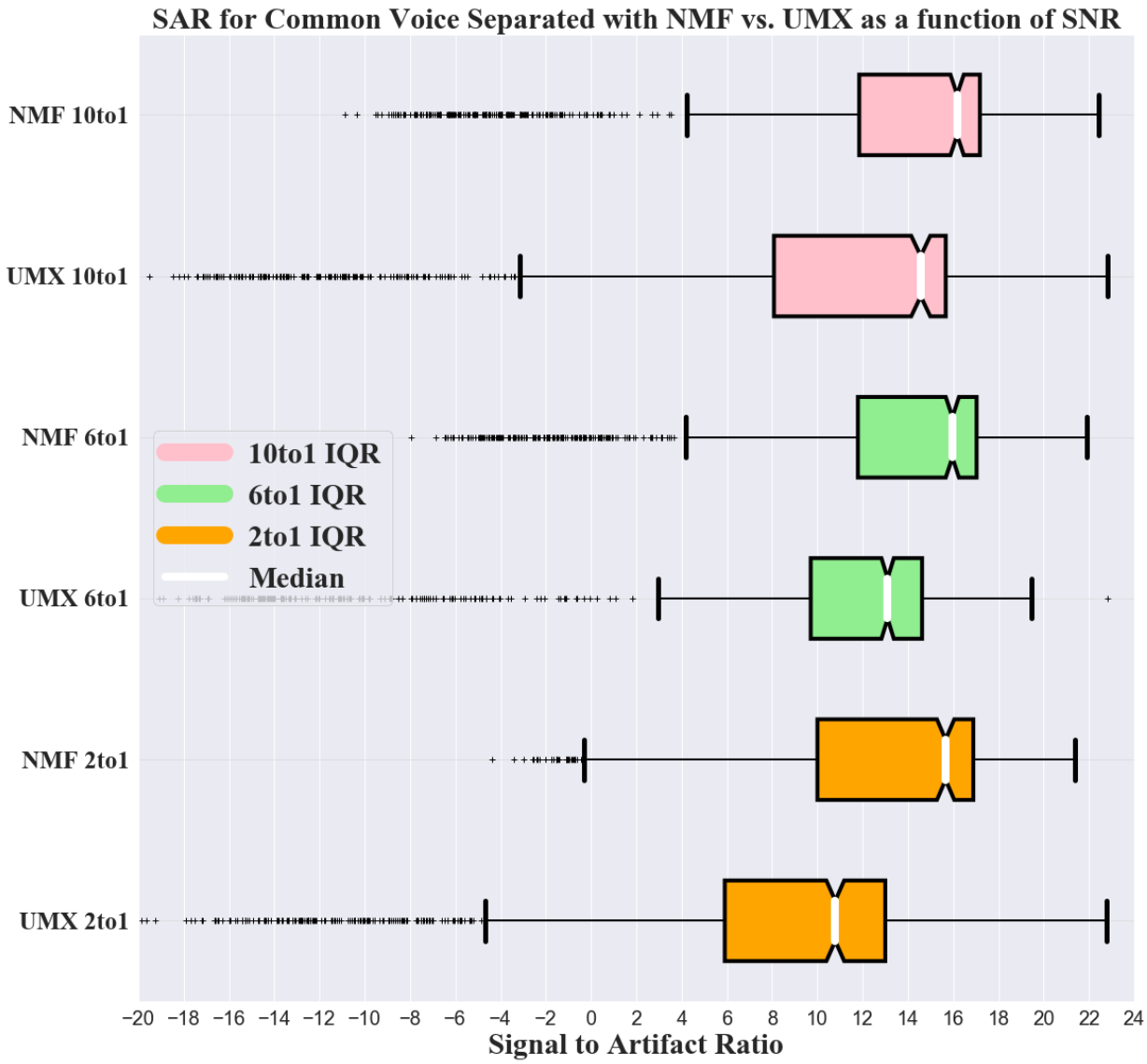

Figure A.3 Common Voice signal to artifact ratio (SAR) distribution.
SAR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at signal to noise ratios of 2, 6, and 10 to 1.

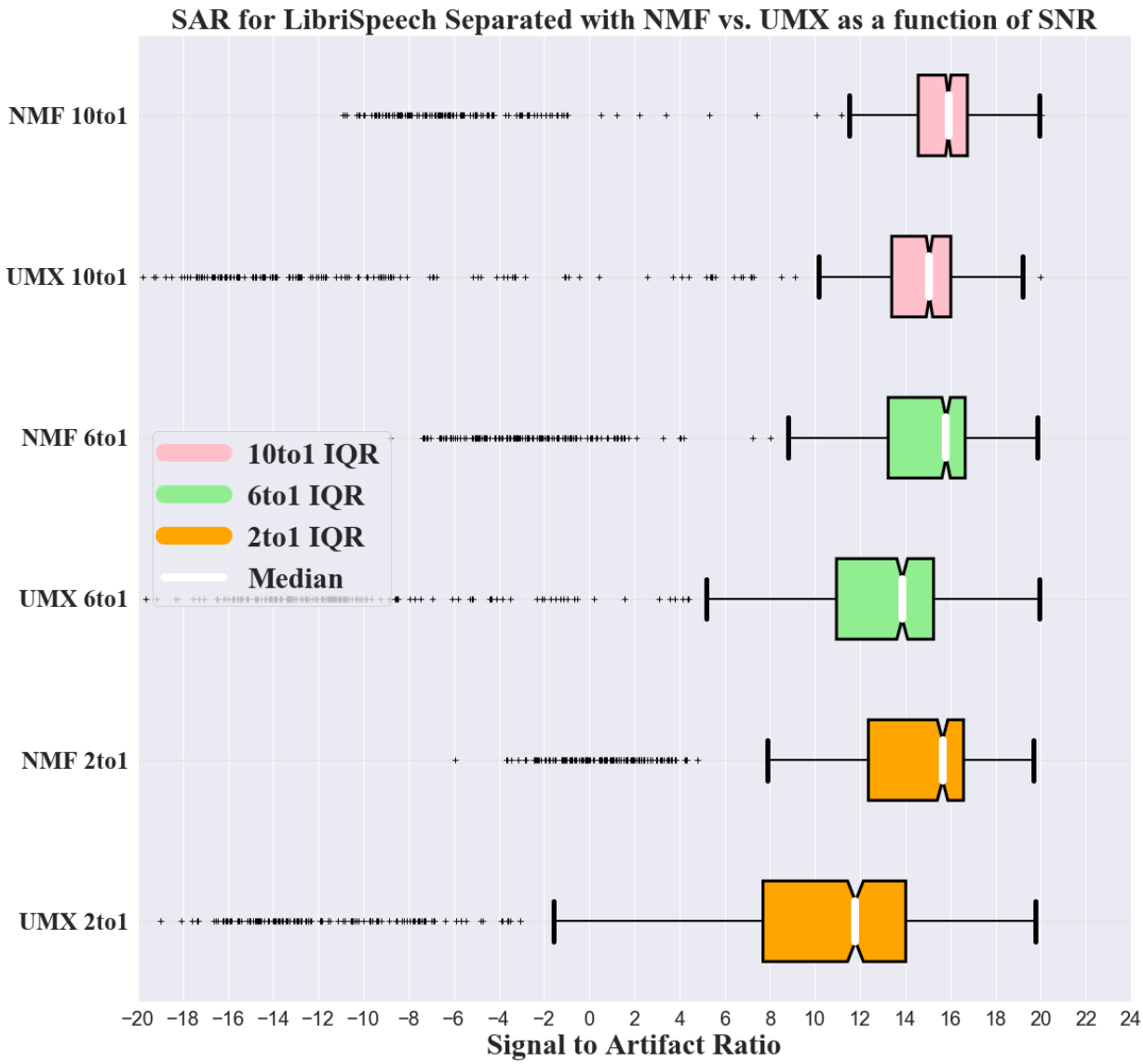

Figure A.4 LibriSpeech signal to artifact ratio (SAR) distribution.
SAR interquartile ranges and medians are shown for Non-negative Matrix Factorization (NMF) and Open-Unmix (UMX) at signal to noise ratios of 2, 6, and 10 to 1.

Appendix B

Spectrograms Examples of Source Separation

This appendix displays the frequency content of 2 clean Common Voice speech examples, and 2 LibriSpeech examples, followed by their spectrograms corrupted at a 2 to 1 signal to noise ratio, and finally their frequency content after separation with NMF and Open-Unmix.

B.1 Common Voice Example 1

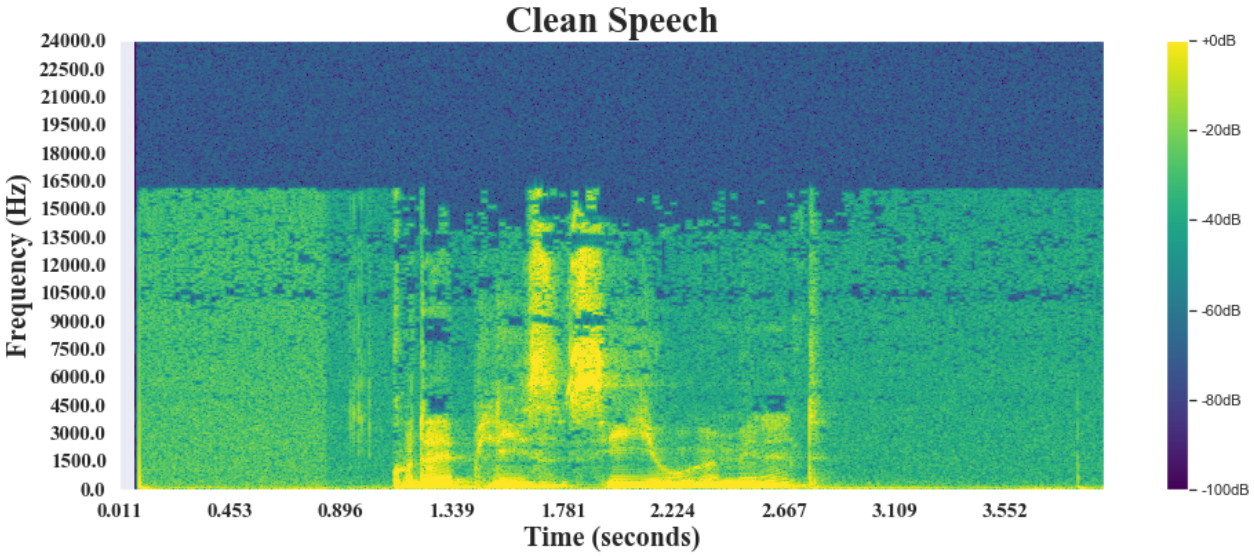

Figure B.1 Common Voice Example 1 Clean

Filename: common_voice_en_596186

Transcript label: “could i please see you a minute”

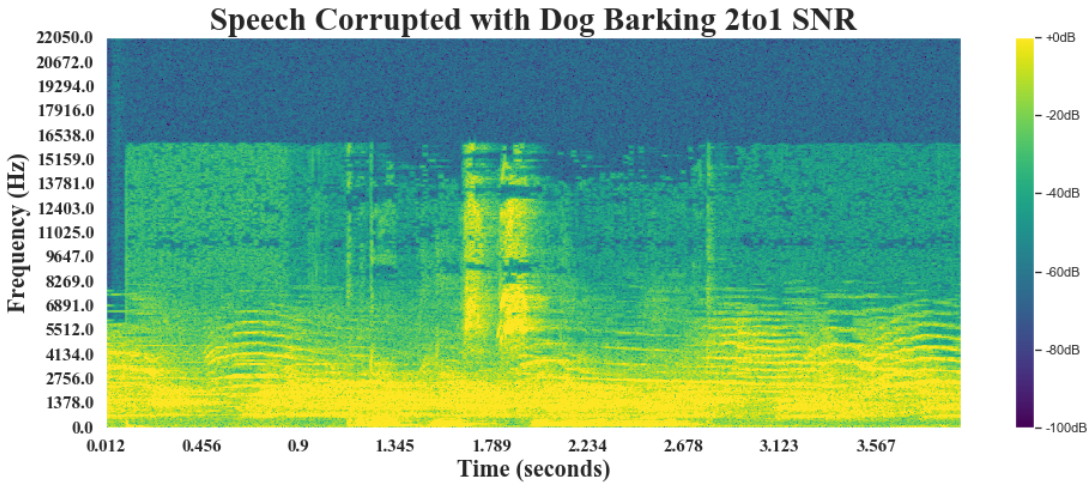

Figure B.2 Common Voice Example 1 corrupted with dog barking at a 2to1 SNR

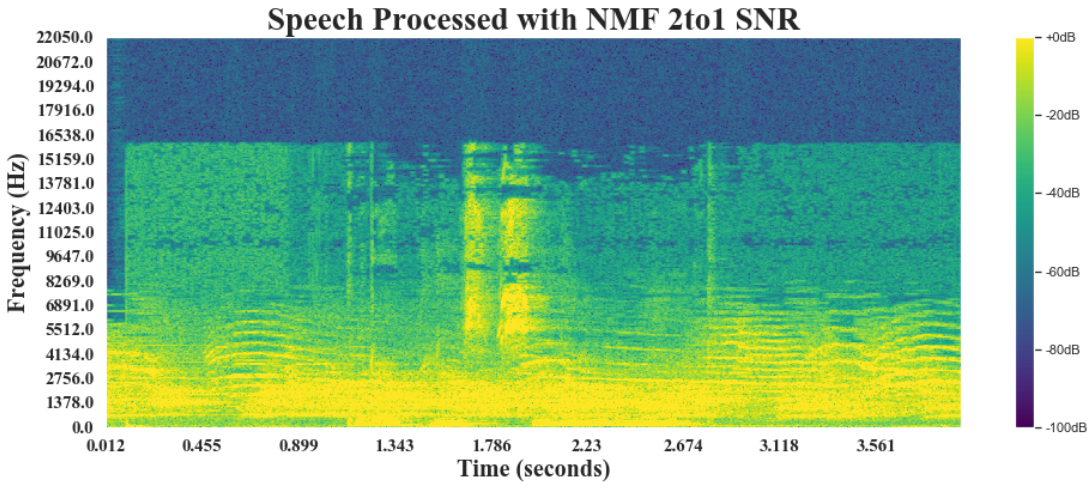

Figure B.3 Common Voice Example 1 processed with NMF

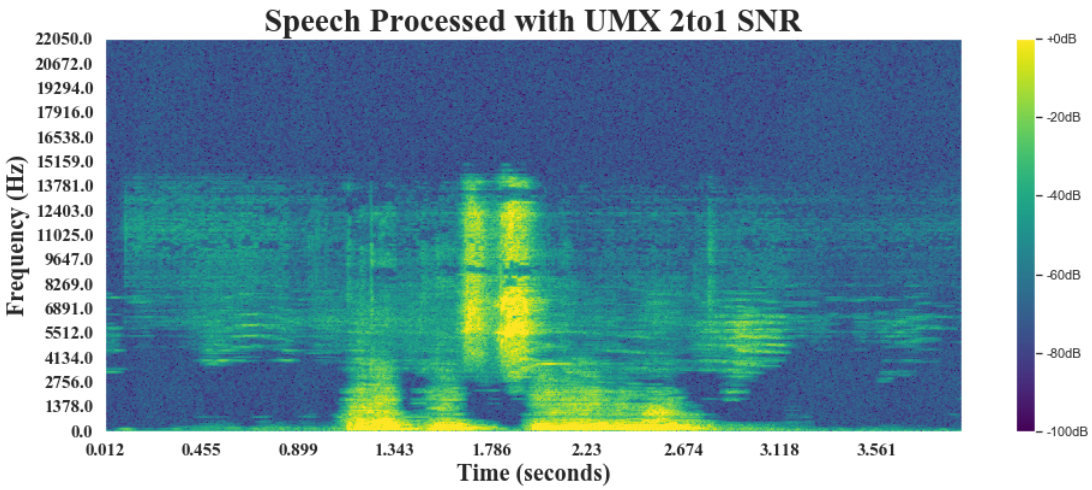

Figure B.4 Common Voice Example 1 processed with Open-Unmix

B.2 Common Voice Example 2

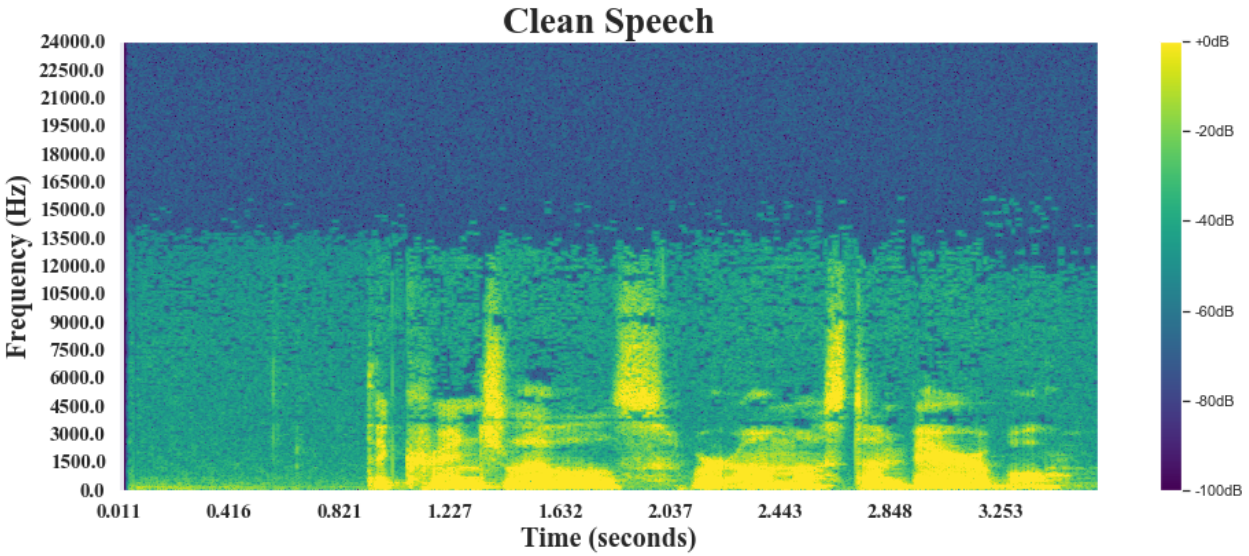

Figure B.5 Common Voice Example 2 Clean

Filename: common_voice_en_614217

Transcript label: “the cringe levels were almost unbearable”

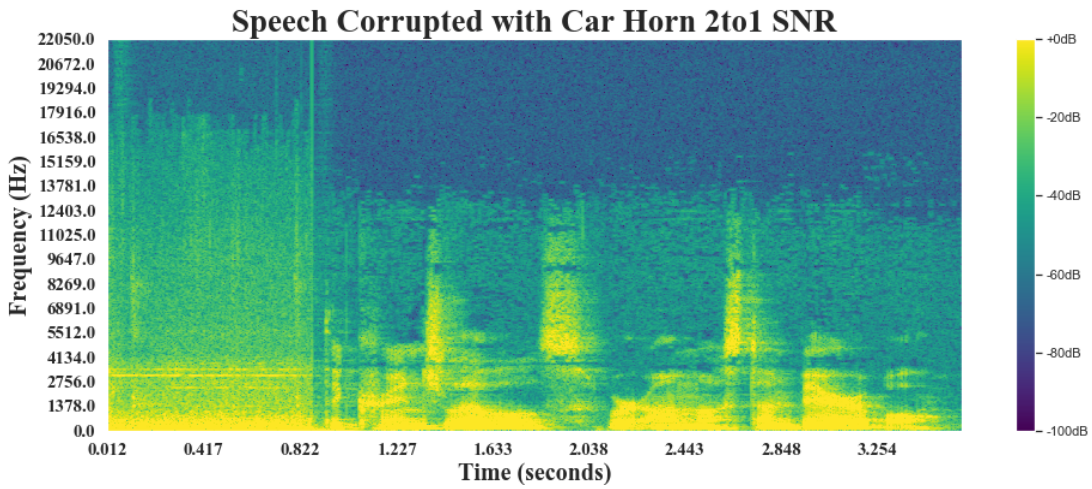

Figure B.6 Common Voice Example 2 corrupted with car horn at a 2to1 SNR

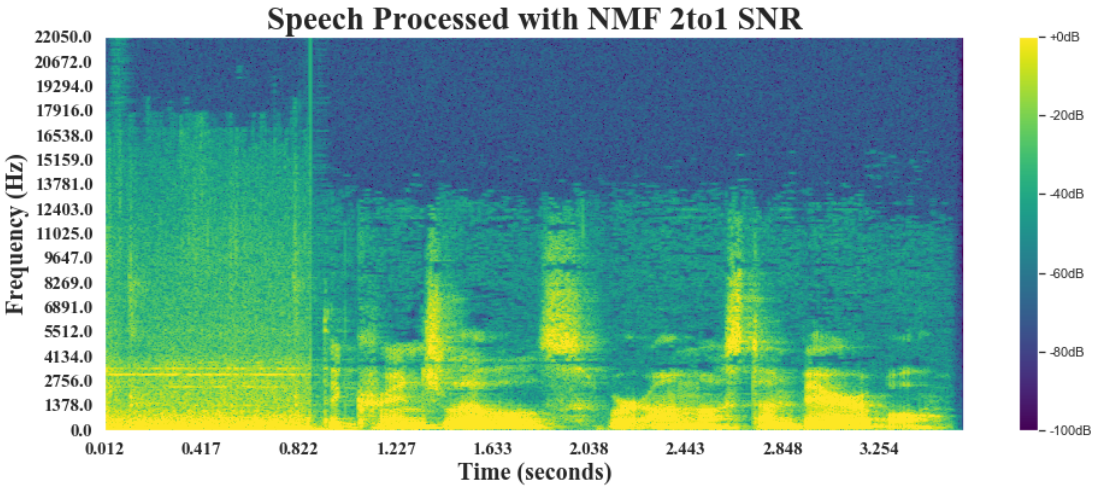

Figure B.7 Common Voice Example 2 processed with NMF

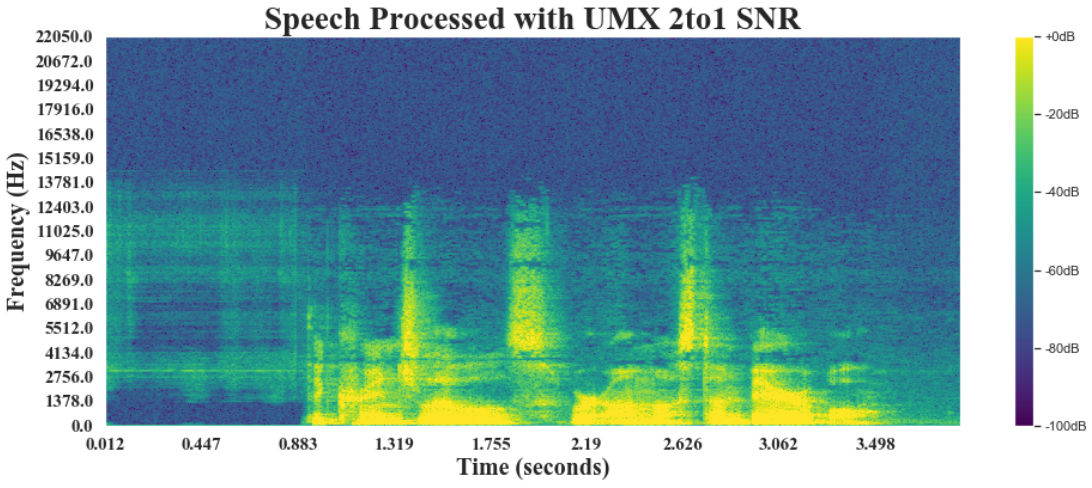

Figure B.8 Common Voice Example 2 processed with Open-Unmix

B.3 LibriSpeech Example 1

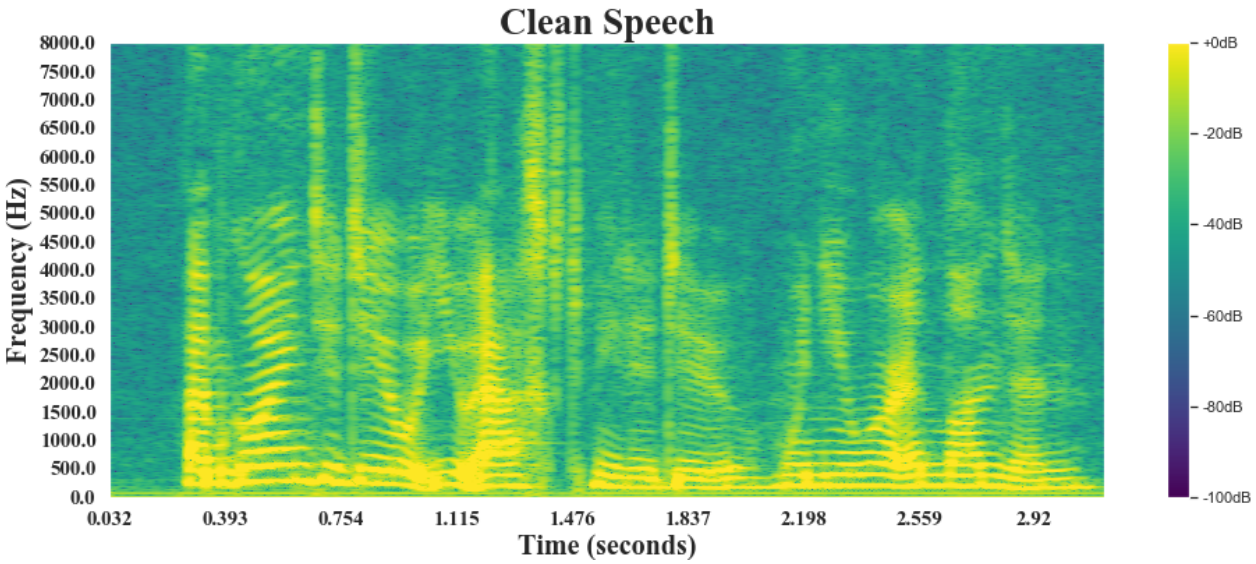

Figure B.9 LibriSpeech Example 1 Clean

Filename: 1462-170145-0017

Transcript label: “i'm going to do what you asked me to do when you were in london”

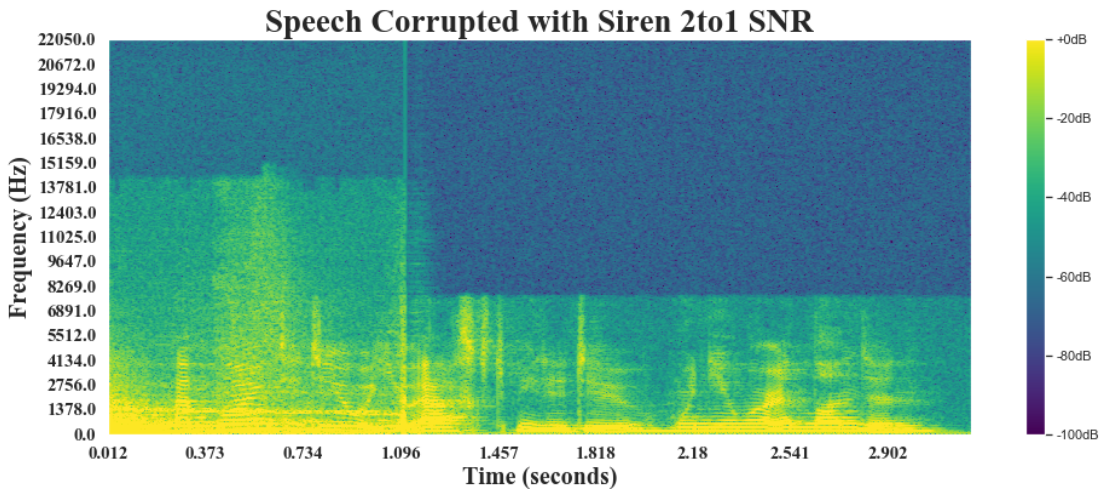

Figure B.10 LibriSpeech Example 1 corrupted with siren at a 2to1 SNR

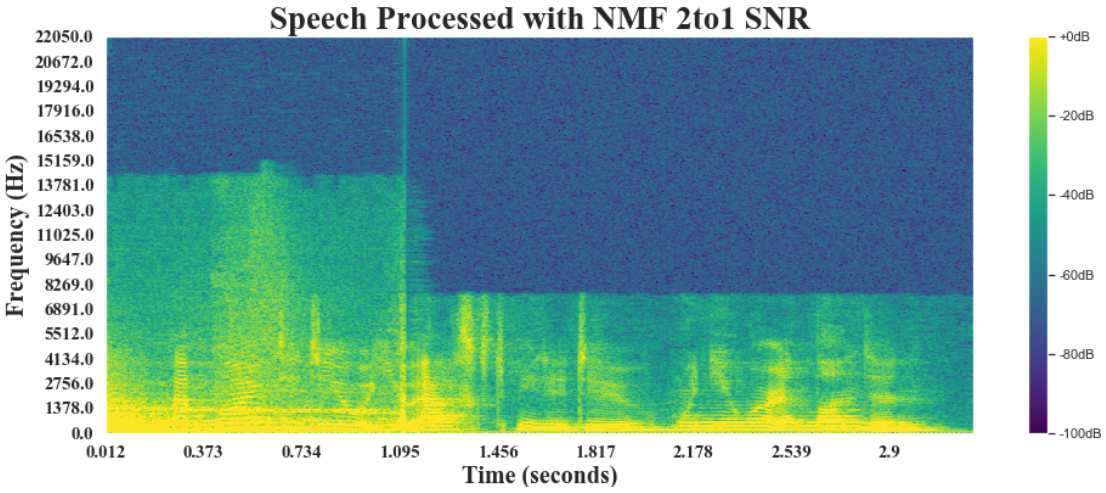

Figure B.11 LibriSpeech Example 1 processed with NMF

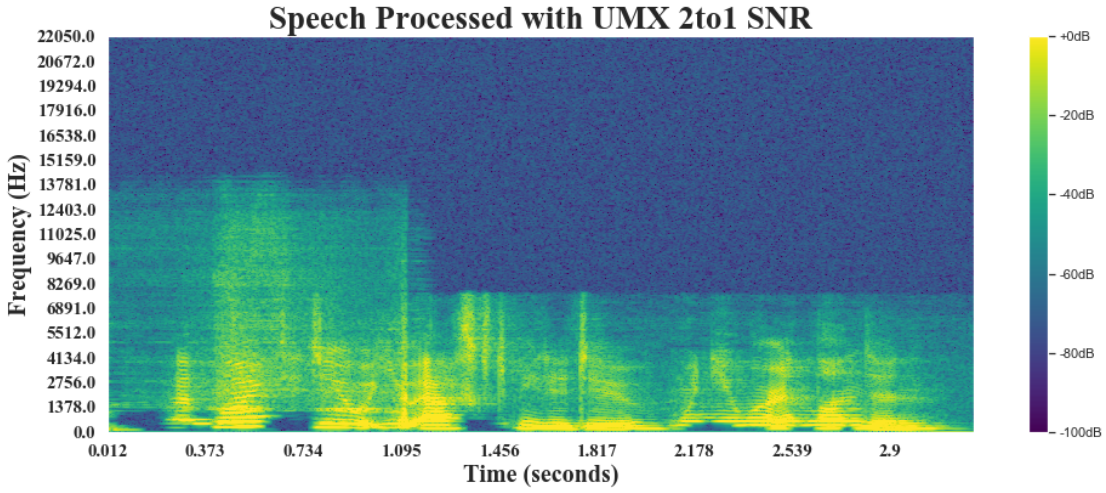

Figure B.12 LibriSpeech Example 1 processed with Open-Unmix

B.4 LibriSpeech Example 2

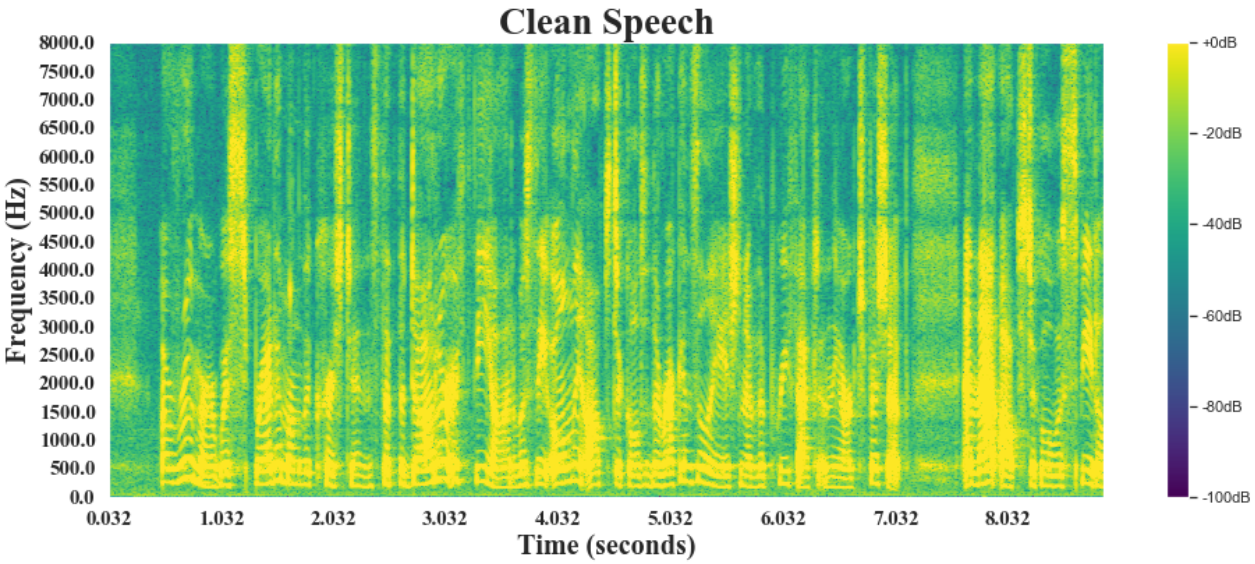

Figure B.13 LibriSpeech Example 2 Clean

Filename: 1673-143397-0005

Transcript label: "a rumor was spread among the christians that the daughter of theon was the only obstacle to the reconciliation of the praefect and the archbishop and that obstacle was speedily removed"

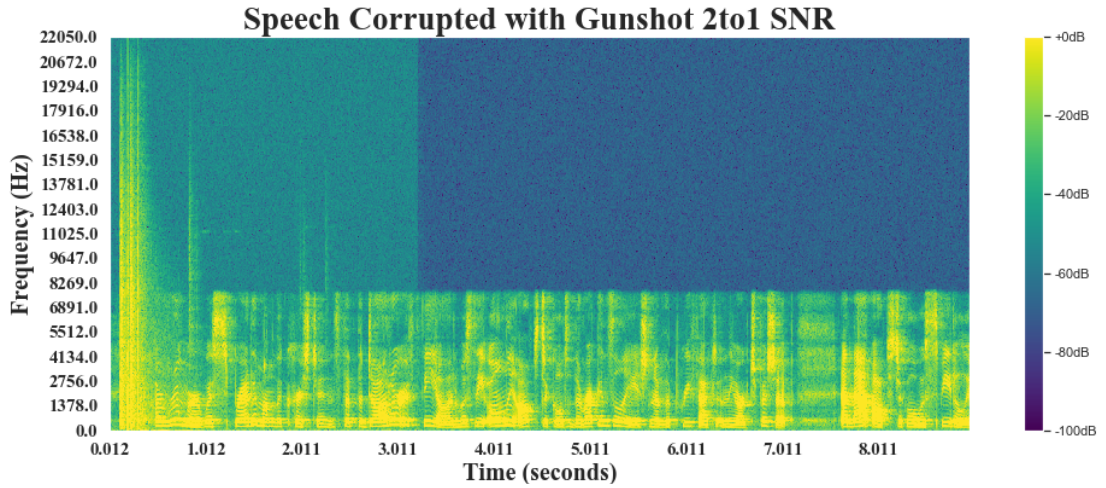

Figure B.14 LibriSpeech Example 2 corrupted with gunshot at a 2to1 SNR

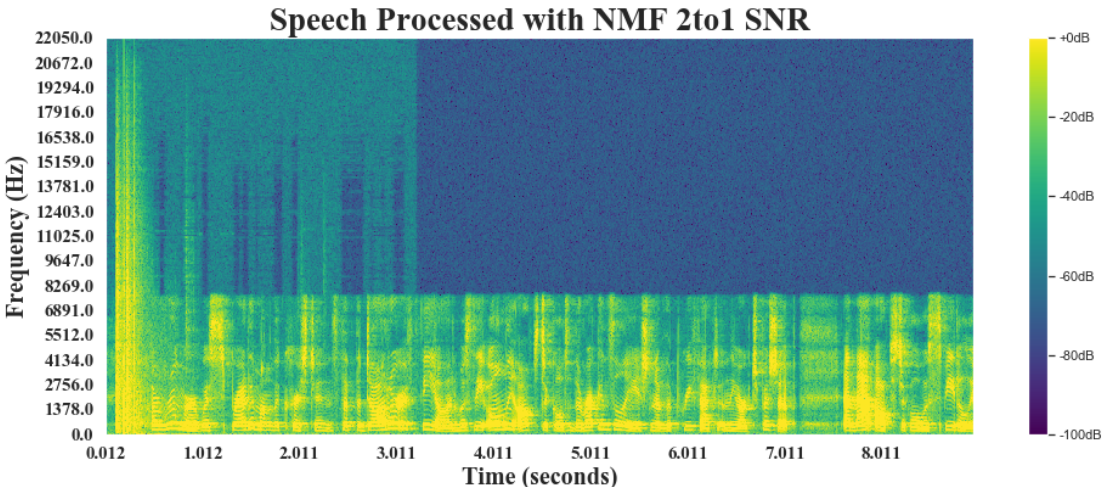

Figure B.15 LibriSpeech Example 2 processed with NMF

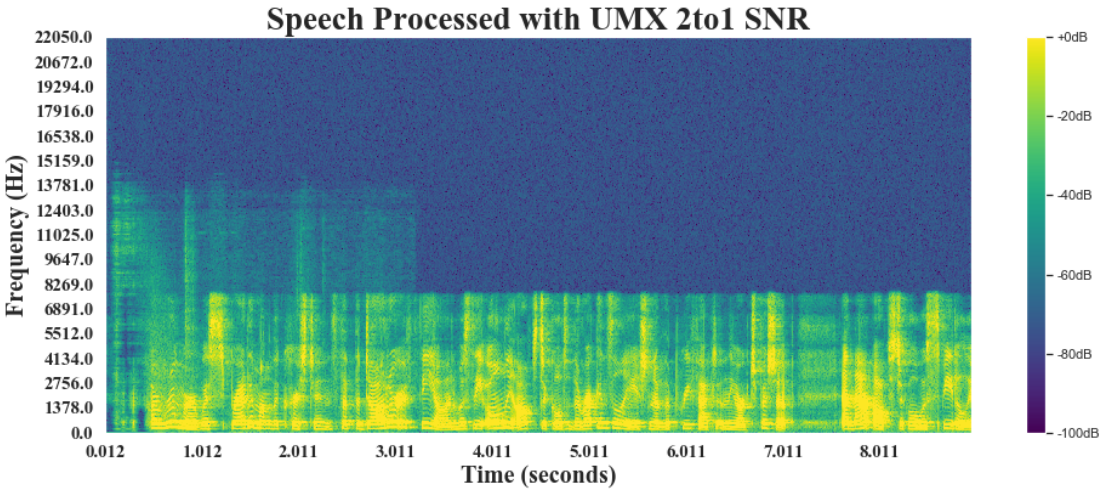

Figure B.16 LibriSpeech Example 2 processed with Open-Unmix