Masking and Reconstructing Speech to Improve Intelligibility

Anders-Petter Ljungquist

MInf Project (Phase 2) Report
Master of Informatics
School of Informatics
University of Edinburgh

2016

Abstract

It is estimated that 1.2 billion people are affected by hearing loss. Often it does not simply make sounds less loud but harder to distinguish, a problem made worse by interfering noise. Though technologies exist to alleviate such problems, these are neither as available nor as effective as they could be. However, the Ideal Binary Mask (IBM) method of speech enhancement has the potential to make noisy speech as intelligible to an impaired listener as to one with normal hearing. Thus, this report presents a system that predicts that mask under severe but typical conditions. It also investigates the possibility of using generative models to reconstruct masked speech and further improve its quality finding that the unprocessed, corrupted phase hampers such improvements. Finally, ways to make use of additional information such as transcripts and speaker identity are explored.

Acknowledgments

I would like to thank my supervisors Dr Hugh Leather and Dr Peter Bell for giving me an opportunity to immerse myself in the fascinating world of speech processing and machine learning. They have provided a mix of radical ideas and grounded opinions, insightful guidance on how to work and knowledgeable advice on what to work on, critical feedback, as well as encouragement that has been indispensable.

Table of Contents

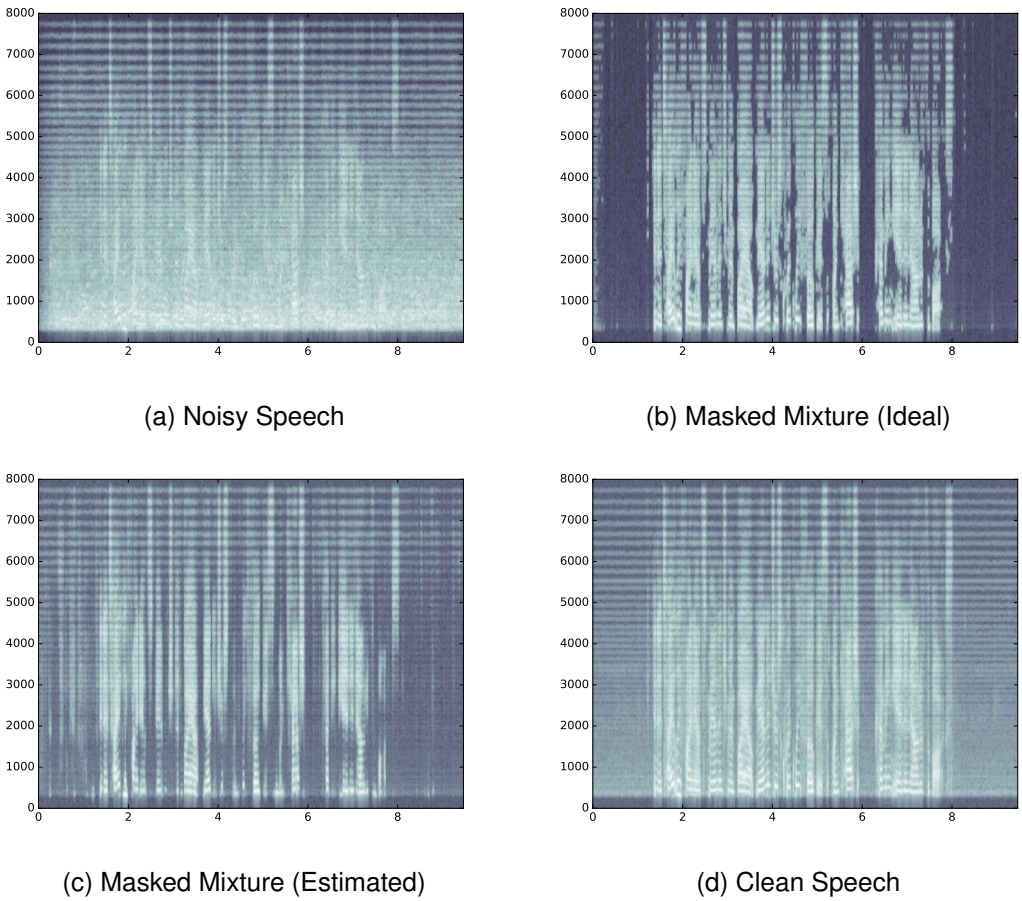
Chapter 1

Introduction

About 1 in 6 people [1], enough such that most of us would at some point have a close relative who is affected, suffer from impaired hearing. Traditionally hearing aid, first analogue then digital, where employed to alleviate this problem. While analog hearing aids were limited to only adjusting the gain of the signal as a whole modern digital hearing aids can do a lot more. Their first advantage is the ability to adjust gain differently in different frequencies and to apply compression algorithms that protect the user from loud sounds as hearing impaired are often as sensitive to loud sounds as those with normal hearing. They are further able to analyze the incoming sound and apply different processing depending on the acoustic context. Such processing could include reducing gain in low frequency bands associated with unwanted noise or increasing gain in frequencies associated with speech, compressing and expanding frequency ranges. Some hearing aids also feature one or several directional microphones that allow them to do spatial filtering [21].

While spatial filtering would address, for instance, the issue of crosstalk it is a feature traditionally associated with high end hearing aids, which can be expensive, though devices attempting to make this more mainstream are entering the market [3] and appear to be well received. Even so a dedicated piece of, possibly expensive, hardware is needed for the task. Spectrotemporal techniques don't need to suffer from this drawback, are applicable also where extreme miniaturization would prohibit the inclusion of multiple microphones and has the potential to further improve the monaural stream obtained from spatial filtering.

With the increasing availability and processing power of smartphones it seems reasonable that they could take on the task of facilitating hearing, eliminating the cost and inconvenience of a dedicated device. Though using the smartphone as an assisted listening device has been done [20] we believe utilizing more prior knowledge of the speakers and machine learning has potential to also transform the ubiquitous smartphone into a potent assisted listening device. This is the premise for this project, but the problem actually addressed in this report ignores any platform, computational and usability constraints that such a deployment would impose. This is motivated by the insufficiency of current techniques to address the problem and the observation that it is usually easier to go from an computationally expensive system that solves the problem



(a) Noisy Speech

(b) Masked Mixture (Ideal)

(c) Masked Mixture (Estimated)

(d) Clean Speech

Figure 1.1: A visualization of noisy speech, speech enhanced by masking and the corresponding clean speech showing the successful attenuation in regions of silence and unvoiced speech. The horizontal banding illustrates the effect of encoding with a Mel spaced filter bank and the gradual changes in attenuation visible in 1.1c illustrate the effect of using a soft mask rather than a hard mask as in 1.1b

to one that is computationally cheap than it is to make a computationally cheap system solve the problem well.

Thus, in this report I present a system that takes speech corrupted with noise and predicts a mask that removes the noise with an average Hit rate minus False Alarm rate (HIT-FA) of 44.6% across novel noise conditions in the targeted range of Signal to Noise Ratios (SNRs). I propose a generative model to reconstruct the masked speech from it context but conclude that with the phase also being corrupted this reconstruction deteriorates intelligibility by reintroducing noise. I also investigate and suggest and extension to this model that uses additional information from transcripts to improve this process.

1.1 Previous Work Carried Out

Speech enhancement by attenuation and amplification can be performed in the time domain, in the frequency domain as well as both simultaneously. Last year the goal of this project was to mask speech only in the time domain. When the interference is not the result of speech, this task becomes similar to voice activity detection. But when the interference can come also from other speakers the task becomes one of speaker identification in noisy conditions. As such focus was on becoming familiar with the field of speech processing and implement a system that can tell who, if anyone, in small group of people is speaking at any one time. But what set the work apart from most published research on speaker enhancement was its focus on noisy conditions and on methods that can work online and in real time.

Similar to this year, the restrictions that would have been imposed by targeting deployment on a mobile platform are ignored. Instead development was done using a mixture of binary routines from the Kaldi framework for speech processing, Weka for modeling, python for various utilities and shell scripts to all components together. The implemented system performs this task and achieves a Diarization Error Rate (DER) of 25.6%, where DER is a measure of what proportion of time is correctly assigned to a speaker.

Because previous work focused on a different approach to the problem and used either different tools or the same tools in a different way, none of the work done, apart from the knowledge accumulated, is reused in the system presented in this report.

1.2 Objectives

Whereas the goal of previous work was to mask speech in the time domain only, that approach is likely always going to be insufficient. Similarly, as we know from hearing aids, simply applying time invariant gain is also insufficient. In order to improve speech intelligibility one likely has to adjust the gain in both domain simultaneously and that is the focus of the work presented in this report.

Thus the goal of this the presented work is to implement a system that given a stream of speech corrupted with noise produces an audio stream of enhanced speech such that it is more intelligible than before. The noise will be from sources that are commonly encountered such as cars and factories with emphasis on those that are the most troublesome for hearing impaired listeners such as restaurants and competing speakers. It will further emphasize SNRs in the range -5dB to 15dB as these are typical for interactions in many environments.

However, speech enhancement is a hard problem and humans likely make use of information not available in the audio stream such as lip movements to recognize speech in harsh environments [28]. Recognizing this this, two relaxations will also be explored. One assumes that the transcript of the speech is known to the system. The other assumes that the speaker is known to the system.

1.3 Organization

The rest of this report is organized as follows;

Chapter 2: Related Works introduces the field of speech enhancement. It provides an overview of the field and its development by first describing traditional methods before proceeding to describe to new methods and the supervised learning framework that enabled them. Finally it briefly recounts the history and current state of method used to evaluate the quality and intelligibility of speech.

Chapter 3: Implementation describes the presented system and the process though which it was implemented. It describes the way in which existing techniques, tools and data have been employed to develop the presented system and and motivates design choices by reference to earlier research and to experiments contained in this report.

Chapter 4: Evaluation assesses the performance of the presented system and its various parts. It introduces experiments to evaluate various aspects of system performance as well as the applicability of these results to new contexts. The chapter presents the results of these experiments and interprets them.

Chapter 5: Conclusion summarizes achievements of the report with reference to key results, highlights important flaws and makes suggestions for future work and improvements.

Chapter 2

Related Works

This chapter introduces the field of speech enhancement. It first describes a set of methods believed unable to improve speech intelligibility. I call them traditional messages. It then describes a set of methods that have recently become popular with the rise of supervised machine learning, as well as the the process of preparing speech to be modeled, typically called feature extraction but here called encoding to call attention to the fact that the process can be reversed. It also describes how various machine learning models are applied to the presented enhancement methods. The chapter ends with an account of how the quality and intelligibility of speech is measured and how these techniques have developed.

2.1 Traditional Enhancement Methods

Many methods have been devised for the purpose of speech enhancement. These tend to fall into one category or another, although many recent methods are harder to categorize. While older methods such as spectral subtraction-, Wiener filtering-, statistical- and subspace-methods are, under the right circumstances, able to improve speech quality, none is able to improve speech ineligibility [14]. In fact, the variations of the methods that show the greatest improvements in quality tend not to be the ones that best retain intelligibility [13]. The exception is where hearing impaired listeners are concerned and at least one study found that ineligibility can be improved.citeneeded. Unfortunately, the effect for hearing impaired listeners is often not studied explicitly and it is not always clear how objective measurements correlate with either subgroup specifically. Therefore, most of the time, I will not make the distinction. At one point, this lack of progress in the past 30 years made it clear that a paradigm shift was needed [13]. As such, this section will only briefly discuss the above mentioned methods before moving on to methods that do improve ineligibility.

2.1.1 Spectral Subtractive

The spectral subtractive methods are among the simplest and the oldest and many variations exist. The basic idea is to estimate the noise spectrum and subtract it from the mixture spectrum. The simplicity of the methods rely on two important assumptions. The first is that noise is additive, that is $y(n) = x(n) + d(n)$ where y is the mixture, x is the signal and d is the noise or similarly in the frequency domain $Y(\omega) = |Y(\omega)|e^{j\phi_y(\omega)} = X(\omega) + D(\omega)$ where ϕ is the phase. The second is that the noise is stationary or slow moving such that estimates made when no speech is present are still relevant some time later when the speaker speaks. Now, given an estimate of the magnitude noise spectrum $|\hat{D}(\omega)|$ we can estimate the speech spectrum as $\hat{X}(\omega) = [|Y(\omega)| - \hat{D}(\omega)|]e^{j\phi_y(\omega}$ [13].

When the noise spectrum is indeed stationary and can be accurately estimated, spectral subtraction can be expected to work well. However, this is rarely the case and large variances in the noise spectra and the estimates thereof tend to cause random, isolated peaks in the estimated signal spectrum that when re-synthesized is perceived as musical noise. This artifact is introduced also when the noise magnitude is estimated to be greater than than the mixture magnitude. In such a case the estimated signal magnitude would be negative a problem that can most simply be solved by half-wave-rectification, the non-linear nature of which again introduces random peaks in the spectrum [13]. Though many methods of addressing these issues exist and alleviate the problem, musical noise remains a problem for spectral subtractive methods.

Note also that in the above equation the estimate of the clean signal does nothing to address the possibility that the phase may also be noisy. At SNR > 5 this is less of a problem but at SNR < 0 it gives the re-synthesized a roughness that affects the quality negatively [13].

2.1.2 Wiener Filtering

Another class of methods that have been around for a while, perhaps because they are well known in general signal processing, is Wiener filtering methods. The basic principle is that because the mixture signal and the target signal are correlated and because the target signal is correlated with itself, the any given sample $\hat{d}(d)$ can be estimated as a weighted sum of the mixture $\hat{d}(n) = \sum_{k=1}^{M-1} h_k y(n-k)$ where $\{h_k\}$ are filter coefficients. This sum can in theory be either infinite or finite and it can cover either all samples for offline operation or just samples already observed for online operation. Interestingly, when the target signal and the mixture signal are the same, the Wiener filter becomes a linear prediction filter and the coefficients linear prediction coefficients [13].

The problem is now to find the best filter coefficients. In Wiener filtering the best coefficient are taken to be the coefficients that yield the lowest mean squared error, which is why the algorithms, unlike spectral subtraction algorithms, are said to be optimal in the Minimum Mean Squared Error (MMSE) sense [13].

One problem with Wiener filters is that they rely on both on the assumption that all signals are stationary but also that signal and noise are uncorrelated [13]. Thus they are more suitable for removal of additive Gaussian noise and less suitable for highly non-stationary noise such as babble and reverberation.

2.1.3 Statistical

Statistical methods are similar to Wiener filtering in that they start with defining an error and the signals before proceeding to analytically find the solution that minimizes the error by equating the derivative of the objective function to zero and solving for the target parameters. But rather than finding filter coefficients that minimize the MSE statistical methods typically aim to find the speech signal that maximizes the likelihood of observing the mixture $\hat{\theta}_{ML} = \arg\max_{\theta} p(\mathbf{y};\theta)$ where y is mixture and theta is signal. Other objective functions, such as Maximum A Posteriori (MAP), are also used. This probability is often modeled using a normal distribution with some variance and the clean signal θ as the mean and some variance [13].

2.1.4 Subspace

The fourth class of traditional methods are subspace methods. The assumption here is that the speech signal and the noise signal occupy different subspaces of the overall mixture space. If this is the case, we should then be able to decompose the mixture into signal component and a noise component and use only the signal component to re-synthesize the speech. The approach has its roots in linear algebra where orthogonal matrix factorization techniques such as singular vector decomposition and eigenvector-eigenvalue factorization exist and help solve this problem [13].

However, subspace methods were developed to remove white and colored noise [13] and are less effective with non-stationary noises such as babble. In fact, seeing how babble or even cross talk noise are produced by similar sources, that is people speaking, it's easy to envision how the speech and noise would not occupy different subspaces impeding the decomposition.

2.2 Supervised Learning Methods

Recently, especially with the resurgence of Deep Neural Networks (DNNs), supervised learning methods are becoming popular. They can broadly be categorized as masking based and mapping based. Masking based methods try to estimate a mask that can then be combined with the noisy spectrum for an estimate of the clean spectrum whereas mapping based methods try to directly map from the noisy spectrum to the clean spectrum [24]. The difference between the two is subtle, at least once has a mapping based system explicitly contained a masking based subsystem and there is a substantial overlap in both feature extraction and modeling. This section will discuss

some of the unique aspects of each approach before exploring how both are done in practice in terms of feature extraction and modeling.

2.2.1 Masking Based

Masking based methods can be distinguished by what the target mask is. One popular and particularly interesting mask is the Ideal Binary Mask (IBM). This methods has also been called channel selection selection since the idea is to break the mixture into discrete channels and at each time step make a binary decision based on some criterion on whether the information in that channel at that time will be allowed to influence the re-synthesis of the estimated clean speech. Several findings motivate the adoption approach. For instance analysis of the gain function yielded by an ideal Wiener filter shows that the values it takes are distributed in a bimodal way with peaks at around 0dB and -27dB. This resembles the effective gain function of a binary mask. It has also been shown that soft gain functions are incapable of improving overall band SNR making it unlikely that they would improve ineligibility [13].

Speech enhancement with ideal binary masking can roughly be split into four components; the decomposition into channels, the criterion by which to choose channels, the estimation of the mask representing these decisions and the re-synthesis of the masked signal. One benefit of this division is that questions about channel decomposition, decision criterion and re-synthesis can all be asked and answered independently of the mask estimation making the process of avoiding bad implementations much cheaper.

One such question is what selection criterion should be used to compute the IBM. The most frequently used criterion is the SNR criterion. It simply states that the channel should be selected if the SNR is above some threshold called the local criterion. The estimated signal is in the frequency domain is thus

$$\hat{X}(k,t) = \begin{cases} Y(k,t) & \text{if } SNR(k,t) > \text{LC} \\ 0 & \text{otherwise} \end{cases}$$

where $\text{SNR}(k,t) = 10\log\frac{X(k,t)^2}{N(k,t)^2}$. In fact, it has been shown that this criterion will maximize Articulation Index (AI) [14], a metric that predicts speech ineligibility well [10] and which I will discuss in more detail later. In order to use the SNR criterion one needs to decide what local criterion to use. Ideally, the local criterion should depend on the mixture SNR as too high a value at too low a SNR will result in no channels being selected which is likely to be more detrimental than choosing some very noisy channels. However, as LCs in the range -10dB to 5dB all produce similar ineligibility, a fixed value in that range is typically chosen [13].

Other selection criteria have also been explored, for instance SRR was proposed to deal with reverberation. Its authors suggest that in purely reverberant conditions the SNR criterion is inappropriate and finds that it compares favorably to spectral subtractive algorithms [5]. However, their definition of SRR as $\text{SRR}(k,t) = 10\log\frac{X(k,t)^2}{R(k,t)^2}$ where $R(k,t)$ is the reverberation magnitude in band k at time t and lack of comparison to the SNR criterion suggests they are simply using a very strict definition of noise as

interference that is statistically independent of the signal and repackaging the SNR criterion for use with reverberation. Nonetheless the paper finding support the use of binary masking.

A flexible selection criterion also facilitates slightly changing the objective of the system. One such changed objective may be speech separation where a criterion based on the ratio between the magnitude of two different speakers can be used [22].

Another question that can be asked is how well this method could potentially perform in terms of how it helps listeners. One study shows that with a local criterion of around 0dB the percentage of correct word identifications increase from 0-20% without masking applied to 90-100% when the ideal mask is employed. With less a restrictive LC, the performance increase falls of until with a sufficiently loose threshold there is no improvement [2]. Another study finds that by employing IBM and a personalized gain function helps hearing impaired listeners achieve the same level of performance as normal hearing listeners. It measured the speech recognition threshold which is the SNR required to achieve at least a certain intelligibility score, typically 50% and found that this was 2.5-7dB higher for hearing impaired listeners under Speech Shaped Noise (SSN) but not significantly different once enhancement had been applied [25].

We can also explore how well one needs to estimate the mask in order for it to be useful. It turns out 25% of binary decisions can be reversed at random without substantially decreasing ineligibility [7]. It has also been shown that the performance in lower frequency channels is relatively more important [12] [25], something that has long been exploited in speech recognition through the use of Mel frequency cepstral coefficients (MFCCs). Further we now know that amplification errors are more disruptive than attenuation errors [26] which in ideal binary masks are represented by false positives and false negatives respectively. Indeed, the HIT-FA metric which can be computed directly on the binary mask and has been found to correlate well with speech ineligibility correlates even better when weighted as $\alpha * \mathrm{HIT} - (1-\alpha) * \mathrm{FA}$ with $\alpha = 0.3$ [9]. When the mask estimator is a DNN we can even optimize the model directly to maximize this objective function, yielding better ineligibility than with traditional error functions [26]. An interesting observation to make at this point is that the binary masking frameworks help us address what was one of the major problems in spectral subtractive algorithms, namely musical noise in the form of amplification errors.

There are many other good questions that can and have been asked and I will explore some of them. However, because the implications of their answers affect also other masking methods and direct mapping methods and in order to keep this section somewhat brief, I will discuss them separately later.

Despite claims that soft masks should be inferior to binary masks [13], others find that when re-synthesizing speech with a binary mask it's beneficial to the speech quality not to binarize the mask but rather use the classifier output directly, which is in the range $[0,1]$, directly [26]. Though seemingly a discrepancy, the first statement addresses ineligibility while the second addresses quality. In fact, when using the corresponding soft mask, called the Ideal Ratio Mask (IRM), as training target it has been shown that the ineligibility produced is comparable to that from the IBM but the quality is

noticeably higher [26]. Similar results have been obtained in speech recognition where use of the IRM has been shown to yield superior performance in a missing feature framework [18]. Though the definition of the mask varies slightly it is a soft mask with values in the range $[0,1]$ that can be interpreted as the probability of the corresponding channel satisfying the section criterion [18] [27].

Other possible masks or training targets exist and have been compared. Target Binary Mask (TBM) is essentially IBMs but where the masking signal is assumed to be SSN and Fourier transform mask (FFT-MASK) predicts the ratio of energy in the signal compared to the mixture. Indeed TBM performs similarly too the IBM with the IRM achieving similar intelligibility but better quality. The FFT-MASK in turn yields performance similar or only slightly better than the IRM [26].

2.2.2 Mapping Based

Whereas masking based methods try to predict a mask that has been designed and the properties of which can somewhat be controlled, mapping based methods try to directly map from the noisy input to the clean output spectrum. The idea is simple but in practice it turns out it may be problematic with direct prediction of spectral magnitude having been shown to consistently yield inferior performance compared to predicting a soft mask. The performance yielded by the mapping target was lower even than the baseline subspace method. With a soft mask defined as $\frac{S(t,f)}{Y(t,f)}$ the estimated signal becomes $\hat{S}(t,f)\frac{S(t,f)}{Y(t,f)}Y(t,f) = S(t,f)$ and perfectly estimating the mask will thus yield the exact same result as perfectly estimating the clean signal directly. One issue with direct mapping is that in practice the target has to be normalized to compress its dynamic range and it seems that the most effective compression is in fact that which is designed into masks [26]. Another potential criticism of mapping based methods, as well as traditional methods such as spectral subtraction and Wiener filtering, is that simple error functions such as Mean Squared Error (MSE) don't necessarily make sense when comparing speech signals. Examples of why this is the case include the masking effect, the relatively greater importance of accuracy in the lower frequencies and the asymmetry between amplification and attenuation errors.

Many other mapping based systems exist that typically use the log-power spectrum as target and compare it favorably to alternative methods. However, as metrics used for evaluation tend not to include one that correlates well with ineligibility such as Short-Time Objective Intelligibility (STOI) and baselines tend to be limited to traditional methods [15] [8] [29] it is hard to establish whether these finding support or contradict the claim that that masking targets tend to work better. One study does evaluate the performance using STOI and compares it to the performance of an oracle IBM, naturally finding that it improves the results but not as much as the IBM [4].

One interesting proposal that blurs the line between masking based and mapping based methods combines feature extraction, mask prediction and re-synthesis into a single DNN that directly predicts the time-domain signal. While its performance is comparable and sometimes superior to masking based systems, it consistently falls short for

ineligibility score [26].

2.2.3 Encoding

Just as important as choosing a good target is choosing a good encoding of the input or set of features. A good input feature is one that contains all the information needed to make a prediction and is structured in such a way that it can be modeled. In speech recognition which has traditionally been based on Gaussian Mixture Models (GMMs) and which require the input to not be correlated for efficient training this has led a widespread use of MFCCs. Despite the connection between MFCCs in the cepstral domain and supervised targets in the spectral domain being hard to imagine, especially once additive noise is considered, and that to my knowledge they have not been used in any proposed system it is possible to use them [26]. A more popular feature in speech enhancement is the Amplitude modulation spectrogram (AMS) which was used together with a GMM to report great achievements [9], that have been heavily criticized for using too similar training and testing conditions [16]. In terms of traditional features, the most useful approach seems to be stacking multiple kinds of features such as Relative Spectral Transform - Perceptual Linear Prediction (RASTA-PLP), AMS and MFCC plus context [19]. A similar combination of AMS, RASTA-PLP with deltas, MFCC and pitch with first and second order deltas can achieve up to double the performance of AMS alone under certain conditions [26].

Now popular model DNNs does not rely on any assumption of the input not being correlated and so attention has shifted away from manually engineered features. Indeed one of the strengths of DNNs is their ability to learn features. As such it is now typical to directly use the magnitude spectrum, or more likely the log transform thereof [4] though some do not explicitly specify this [22] [23], the power spectrum [29]. Semi-engineered features such as Mel-filterbank features [15] [8] and Gammatone features [25] are also common and strike a balance between utilizing prior knowledge about human hearing and automated feature learning.

One benefit of using filter bank features is that the information density is increased. A raw Fast Fourier Transform (FFT) spectrogram typically has 128 or 256 coefficient and will when reversed almost perfectly re-synthesize the original signal. However, research shows that much fewer channels are needed to reproduce intelligible speech and many implementations use anywhere 25 to 64 filters to reproduce good sounding speech, substantially reducing the computational load of training and running a system. According to one study, ineligibility after application of IBM starts plateauing already at 16 channels in babble noise at -5dB, or 24 channels if the noise is SSN. The corresponding word identification rate is up to about 60-70% from less than 10% without the IBM [12]. While it falls short of the 90-100% reported with 128 channels [2], it shows that if needed significant computational savings can be made by decreasing the number of channels used.

For temporal context it is typical to use first and second order derivatives with traditional features whereas for spectra and filterbank features that are coupled with models that handle correlation in the input

2.2.4 Modeling

The move in speech enhancement to supervised learning methods coincided with the latest wave of interest in neural networks and most proposed systems will use some variation of a deep graphical model. There are however examples of systems that have used GMMs [9] and Support Vector Machines (SVMs) [26] with questionable and modest success respectively.

When it comes to DNN models, possibly the most popular approach is a plain feed-forward architecture. A common configuration for these is 3 layers of 1024 units each. Though one advantage of Rectified linear unit (ReLU) units is that they reduce the need for generative pre-training, some still choose to do this [11] while others observe that this may not be necessary when ReLU are used with dropout training and sufficient data [26].

Another popular kind of model used exclusively with mapping based methods is the Deep auto encoder (DAE). The main difference compared to previously mentioned architectures is the way the network is build; instead generative pre-training the network is build by training a shallow Auto encoder (AE) with one hidden layer and then training subsequent layers as AEs with the previous hidden activations as input and output until the desired number of hidden layers is reached. One investigation into the optimal size of such a network finds that 3 layers of 100 units each yield the highest quality [15], a surprisingly small network considering the previously noted collective preference for layers of size 1024.

More exotic architectures include Multi-resolution stacking (MRS) whereby ensembles of DNNs are stacked and the input features are fed to the network multiple times [24] and Recurrent Neural Networks (RNNs) in which the some hidden activations at one time step are fed as input in the next time step [8]. The difference from RNNs was found to be insignificant while MRS yielded consistently better ineligibility than masking base methods which in turn outperformed the reference mapping based methods.

2.3 Speech Evaluation

Both for the sake of developing systems and comparing systems developed by different researches it is important to be able to accurately assess performance. A few distinctions can be made when talking about evaluating speech. First there's the distinction between quality and ineligibility, where inteligiblity has been defined as "the probability of correct recognition when context is available" [17] whereas quality is a measure of how pleasant the speech sounds. The second distinction is between subjective and objective measures. Subjective measures are listening experiments and are likely the most accurate but they are time consuming and therefore also expensive which is why an objective measure that can be computed by machines is desirable. Lastly one may distinguish between intrusive and non-intrusive measures, that is measures that either need or don't need access to the clean speech as a reference for comparison [13].

It may also be useful to identify the quantities we are attempting to measure namely distortion, interference and artifacts [17]. Unfortunately it's likely that any mapping from this tuple to a single number will be as informative as the tuple, but there are single number measures that do this, typically with an emphasis on one aspect or the other. Another aspect of the problem of designing an evaluation measure is that depending on the algorithm that is the source of the speech we want to evaluate we may encounter different combinations of distortion, interference and artifacts with different characteristics. This is why it is sometimes telling of a researchers background what evaluation measure they use.

One of the first measures developed to quantify intelligibility in the 1920s [13] is the AI which has been found to correlate well with intelligibility [10] but it's less predictive when the noise is not stationary or the gain function not linear [13] and is only used today in isolated cases [9]. Speech transmission index (STI) too is a function of the SNR in each band and it too has been found to not account well by the kind of distortions introduced by modern speech enhancement. Unlike the aforementioned measures, STOI is based on the correlation between the target and the estimate. With $r = 0.90$ [13] it has among the highest correlations with intelligibility. Another popular measure is Perceptual Evaluation of Speech Quality (PESQ) which despite primarily being a measure of quality with $r = 0.92$ [13] also correlates reasonably with ineligibility ($r = 0.79$). These two, STOI and PESQ have in fact been recommended measures for ineligibility and quality respectively [17] in favor of also investigated Blind source separation (BSS) metrics Signal-to-distortion ratio (SDR), Signal-to-interference ratio (SIR) and Signal-to-artifact ratio (SAR). This despite measures such as Normalized-covariance measure (NCM) having even higher correlation with intelligibility [13], but as no studies I know of use that metric it's hard to fault them.

Chapter 3

Implementation

In order to assist the reader in assessing the results presented in chapter 4 and their implications, this chapter provide descriptions of what has been implemented and why. It first gives an overview of the components of the system and how they interact before discussing each in more detail.

3.1 Overview

The proposed system can be roughly split into six or seven components as illustrated in figure 3.1. The data preparation component takes speech and a specification in the form of noise and SNR and produce corrupted output while the speech evaluation component compares such corrupted speech or enhanced versions thereof to clean speech in order to produce a measure of the speech intelligibility. The encoding component transforms time domain signal to a set of codes that are amenable masking and reconstruction operations while the decoding component transforms these codes back to time domain signals. The masking component predicts a mask indicating what units of code are unreliable and should be removed while the reconstruction component reconstructs the removed units from their context. Finally, the forced alignment component uses the noisy speech and a transcripts to generate an additional stream of information in the form of a phone transcript to facilitate the task of reconstruction.

It is worth noting that the masking and the reconstruction subsystems are implemented in the reverse order in which they would execute, and are thus presented in this order. What allows this to be done is the IBM framework and when developing and evaluating the reconstruction subsystem, the masking subsystem is substituted with and oracle providing the true IBM.

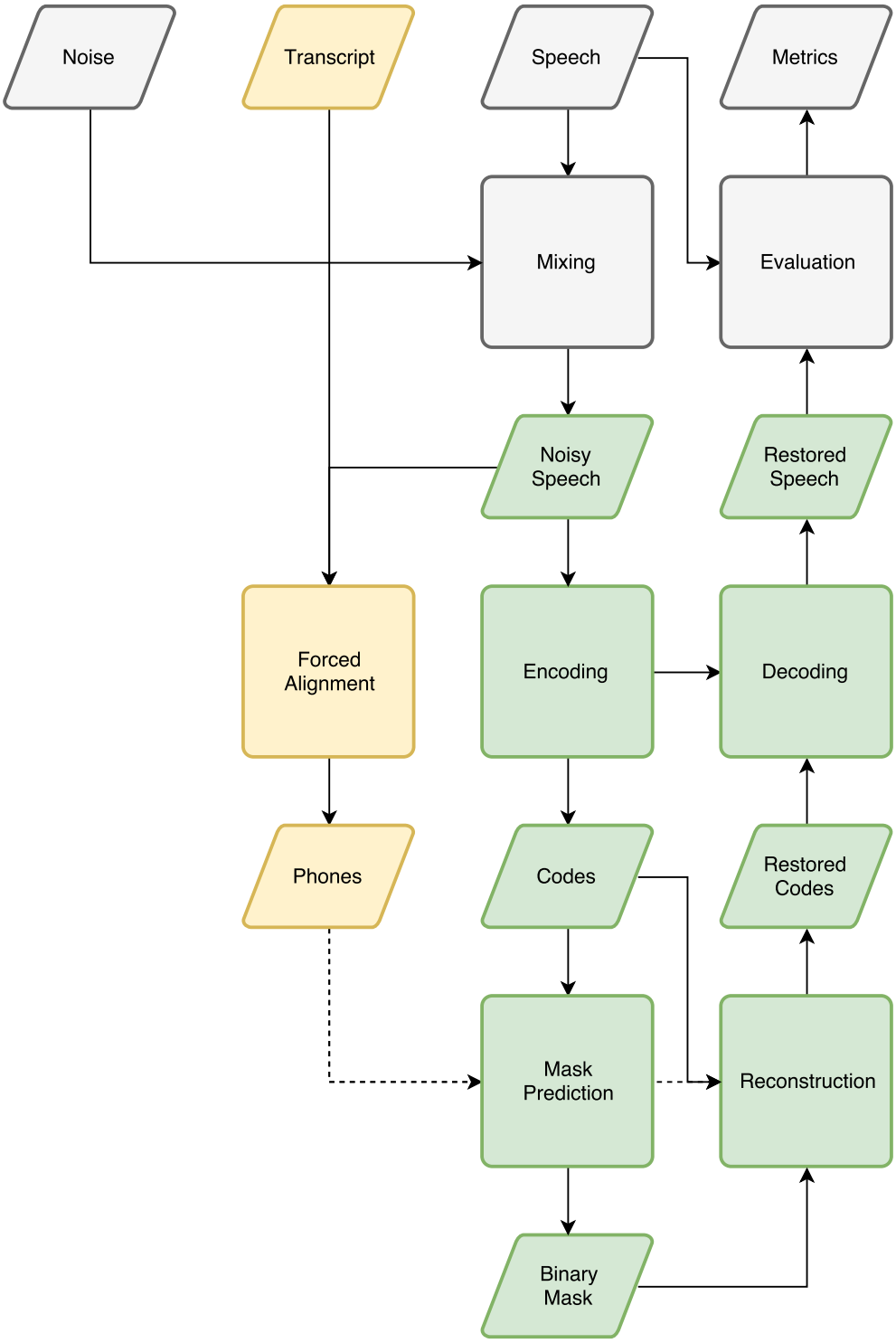


Figure 3.1: An overview of the implemented system and its components with core components colored green, development components gray and optional components yellow. It shows how noise, speech and transcript data is used by the system resulting in a performance score and illustrates the relation between different components as well as what data is passed between them. Dotted lines are used to clarify that the source is 'Phones' and not 'Mask Prediction'.

3.2 Data Preparation

The data used and how it is prepared is an integral part of developing machine learning systems. This section describes what data is needed, how it is obtained and some of the pitfalls of data selection and preparation.

Data is needed for two distinct purposes, for training statistical models and for evaluating the performance of the system. It is not necessary to use the same corpus for both tasks, but in the case where different corpora are used care must be taken to ensure that they come in or are transformed into compatible formats. Notable points of incompatibility are sampling rate and filtering meant to emulate the effects of transmitting speech over the phone. Once compatible, it is theoretically possible to mix corpora even for a single task such as training. However, as combining multiple corpora is rarely done much of the work must be done for the first time. In doing so, one would also need to take great care to intricate details such as re-sampling, filtering, characteristics of recording setup, type of speech and composition of speakers. As such I have opted to use only one corpus at a time.

Having decided to use only one corpus at a time, there are several other considerations to take into account. One requirement in my case is the availability of transcripts, which both TIMIT and WSJ but few others have. Though critics of TIMIT claim it encourages research that does not generalize well, at least two studies [8] [24] similar to this one have used it and could provide good reference points where performance is concerned. It also has phone transcripts which can be helpful for troubleshooting and evaluating phone alignment procedures. Further, both corpora have the qualities that they are English language, readily available on the university file system, includes a variety of speakers and, very importantly, have existing routines for transforming them to the standard Kaldi data format.

The downside of both corpora is that they do not include any noisy speech or methods for corrupting speech. The Aurora series of corpora tries to address this by building on WSJ with a standardized set of noise. I attempt to use it but it is not readily compatible with how my system. Other noisy corpora exist, for instance NOIZEUS [6], which is meant to enable standardized evaluation and comparison of systems developed using heterogeneous data. Unfortunately, it contains only an evaluation set of data and its 8kHz sampling rate coupled with filtering renders. Instead, I opt for using freely available noise samples from the web that I down-sample and convert to mono using Audacity.

Now, conveniently noisy speech can be simulated by mixing clean speech with noise, automatically providing accurate mapping from noisy to clean speech and allowing any large corpus of clean speech data to serve both systems. It also allows a given amount of collected speech and noise to form a collection of noisy speech the size of which can be the product of the size of the speech and the noise corpora and thus orders of magnitude bigger than either. The main drawback of this methodology is that it fails to account for the Lombard effect whereby people raise their voice in noisy environments.

Mixing speech and noise can be done with a third party tool such as `FaNT`. It's a compiled binary that supports mixing a list of signals with a given noise at a given SNR as well as additional options such as applying filters to simulate various transmission channels. However, as it operates on lists of input and output files which it reads from and writes to disk and can thus be cumbersome to include in a Kaldi recipe which make good use of pipes to avoid writing to and reading from disk and to parallelize computation. As such I opted to write a Kaldi tool which mixes signal and noise in the time domain by first calculating the power in the speech instance as well as the noise instance and then adding scaled versions of the two together sample by sample, repeating the noise as necessary when it is shorer than the signal.

$$m[i] = s[i] + \frac{A_s}{A_n} 10^{-\mathrm{SNR}/20} m[i \mod I_m]$$

where m is the output mixture, n is the input noise signal, s is the input speech signal, SNR is the desired SNR, i is the index of the speech signal sample, I_m is the total number of samples in the noise and $A_x = \frac{1}{N}\sum_{n=1}^{N} x[n]^2$ is the amplitude of a signal.

Corpora like TIMIT and WSJ already have designated training, validation and evaluation subsets each of which contain a set of speakers not overlapping with that of any other. This ensures minimal correlation between the sets and I must maintain a similar separation between the sets of noisy speech I create. Failure to do so may result in optimistic evaluation and the effect may be as substantial as reporting 80% HIT-FA when a more realistic evaluation would have indicated 10% HIT-FA [16]. To do this I split the noise into three sets, each of which contain one 30 s segment from each of some noise recordings. The first two sets contain the first and second segments from the same noise recordings while the third contains segments only from entirely different recordings. This lets me evaluate for instance how well a system trained with the noise in one restaurant would perform in the same restaurant but at a slightly later time as well as in a different restaurant altogether.

3.3 Speech Evaluation

The other endpoint of the system is evaluating it output. This is primarily done using the STOI metric and this section describes how it is done and how it compares to some other methods of evaluation.

One side benefit of simulating noisy speech is that we have access to the clean signal we are trying to retrieve and this lets us use intrusive evaluation methods. The simplest example of this is MSE which can be calculated either directly in the time domain, in the frequency domain or even in terms of encoded speech. Because it is cheap to compute, can be computed at various point in the enhancement process and is simple to interpret mathematically it will be used occasionally. It is also already implemented in several of the learning routines available through Kaldi and where it is not available it is trivial to implement. However, as was discussed in 2.3 it is in general not well suited for evaluating speech.

Instead, the metric I will primarily refer to when comparing outputs is the STOI metric. The only existing implementation is available as a function in Matlab. It would be convenient to be able to wrap this function in C++ in order to seamlessly integrate the evaluation into the Kaldi pipeline. However, the Matlab code would neither compile into a C++ library nor to a stand alone command line argument. The easiest way to access the implementation was thus to implement a matlab script that reads a list of files to compare from disk, calls the STOI implementation and writes the results to disk. A shell script then prepares these lists and calls on matlab to execute the script.

Unfortunately evaluating the ouput this way is computationally expensive, taking about 10 minutes for the WSJ validation set of 503 utterances. It is thus impractical to use it to monitor the training or even compute performance over the training data which may contain as many as 37000 utterances and would take an estimated 12 hours to compute. To speed up evaluation and reduce external dependencies of my Kaldi based system, I attempted to implement the metric directly in C++. Though experimentation showed that the output was in the required range $[0, 1]$ and that it ordered samples correctly, absolute values did not correspond to those from the original implementation. Because of this and because this metric will also be the primary way of comparing the performance of my system to that of other systems, I chose to use the slower and more cumbersome calls to Matlab.

3.4 Speech Coding

The first step in training and using a machine learning system is transforming the raw data into a form that can be modeled and in this case also masked. This section describes what approaches I considered, how I compared them to decide which to use and how this coding method was implemented.

The reason coding has to be done at all is that audio in its raw format as a time series of air pressure measurements is traditionally hard to model, though current trends suggest this may be changing, and importantly that binary masking is not applicable in the time domain. Many techniques exist for encoding speech and are discussed in section 2.2.3 but not all can be readily masked and decoded. One class of coders, which I will call filter-bank coders, are simple, proven to work, amenable to masking and allows decoding.

There are two primary ways in which they can be computed, in the time domain by application of finite impulse response filters and in the frequency domain after an FFT transformation. The former method isn't readily invertible but has been used for the purpose of masking as the mask can be applied in the time domain and the channels then simply windowed and added. However, as I attempt also to reconstruct masked units this method of decoding is not viable. Time domain signal processing is also not something that has a lot of support in Kaldi nor something that I have experience with and as such I use the second method.

Now, once converted to the frequency domain and separated from its phase the speech can be encoded by multiplying the matrix representing the signal in the time-frequency

domain with a bank of filters conveniently represented as another matrix. Decoding is then achieved by multiplying the encoded signal by the transposed matrix. Because the coding matrix has many zeros in predictable locations this is a slightly naive and inefficient implementation but this is alleviated slightly by coding not being a very frequent operation and by the ability to perform matrix multiplication on GPU or even implement as a layer in a DNN. The upside of this simple implementation is that the only difference between codings is the matrix.

In terms of designing this matrix, three aspects need to be considered, the shape, spacing and number of filters. As some studies have shown that as few as 12 channels provide high ineligibility while others use all 256 channels from the FFT transformation and many use something in-between, I wanted to investigate the effect of different spacing and shape and the trade off between number of filters and quality.

I prototyped a system in Matlab that encodes the speech using the above approach and a variety of matrices, computes the IBM from noisy and clean speech, applies the IBM, decodes and resynthesizes the speech and finally evaluated the quality using the STOI metric. I try two popular classes of filters, Gammatone and Mel with 4 to 96 and 64 filters respectively and compare them to the quality when using no compression. I also investigate two methods of decoding, one where all energy is assigned to the central frequency of each filter and one where the transpose filter-bank matrix is used, thus assigning energies to all frequencies weighted by how much they contributed to that filter during the encoding. I denote these decoding methods one-to-one and one-to-many respectively.

The results are presented for the TIMIT validation set with restaurant noise at 5dB SNR in figure 3.2. Perhaps the first thing we notice in the figure is that whereas there are 256 FFT channels there are no more than 128 and 64 Gammatone and Mel frequency channels. This is because with if we had any more filters, multiple filters in one bank would have the same central frequency. The conclusion that can be drawn from the graph is that Mel filters are mostly superior to Gammatone filters and reaching performance close to that of no compression with a code size of $\frac{1}{4}$, not counting the phase that is retained unprocessed. We see also that one-to-many inversion is superior to one-to-one inversion when the banks are small but inferior when the banks are big, though with Mel banks this change coincides with the maximum size of the bank. Now, as one-to-many mapping deteriorates with lower SNR and because the difference between the two with 64 filters is likely within the margin of error I chose to use 64 Mel spaced filters with one-to-one inversion and have implemented this in C++ with Kaldi.

A few other observations can also be made for instance it is less clear which of Mel and Gammatone is better when the number of filters are in the range 16 to 32. Curiously this is the same number of filters as is traditionally used in speech processing where some studies have found Gammatone filters to be superior.

Another curious observation is that the gap between the two inversion methods decrease as the number of filters increase, suggesting that the phase carries a lot of information.

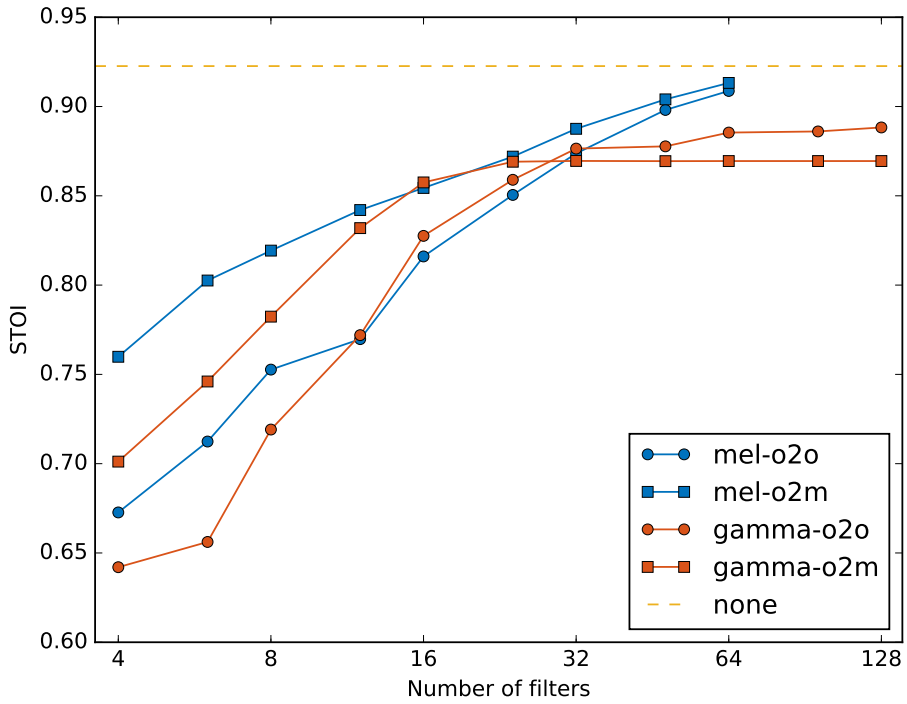


Figure 3.2: A comparison of different methods of encoding and decoding speech. It shows the intelligibility that results from encoding noisy speech using a given method and applying the IBM before decoding. It indicates that the Mel filter bank is superior and that the one-to-one decoding method is preferable when sufficiently many filters are used.

3.5 Code Reconstruction

Speech has a lot of inherent structure and though this can be locally masked by noise and consequently the IBM, patterns stretch both over time and over frequencies and the information gleaned from reliable areas reasonably hold a lot of information about speech in masked areas. Inspired by this observation and recent progress in computer vision where deep generative graphical models are used to reconstruct parts of images, I explore if the same can be done for speech. This section describes the implementation and optimization of the basic model as well as an extension that uses the transcript of speech to improve performance.

3.5.1 Implementing Sampling

The basic model used is the Restricted Boltzmann Machine (RBM). It's an energy based model graphically represented as a fully connected bipartite graph where one of the disjoint sets represent the visible units or input and the other set represent some latent properties of the distribution to be modeled. It's widespread in use thanks to being a prominent way by which to pre-train DNNs and is thus implemented with training routines in Kaldi. However, because the primary use of RBMs is pretraining deep discriminative models, sampling from RBMs and more specifically conditional sampling given some visible units is not implemented in Kaldi. Since the RBM is bipartite and every every visible unit is conditionally independent of every other visible unit given the hidden units and hidden units are similarly conditionally independent as well, sampling from the model can be performed using block Gibbs sampling.

The procedure of block Gibbs sampling is simply to compute the activation probability of the hidden units and then sample their states from these probabilities before similarly sampling from the visible units given the hidden units. These steps are repeated to allow the sampling to burn-in or to reach a point where it accurately represents the distribution. To condition the sampling on some known units, the known units are reset to their original values before proceeding with another round of sampling. Under some circumstances omitting the sampling and directly propagating the probability will allow faster convergence and this is the way I do it. As such, the sampling becomes deterministic and will eventually converge to the most probable state of the unknown units given the known units. When this state becomes stable, I terminate the process. Gibbs sampling itself is simple to implement but finding a good way to interface my implementation with that of Kaldi was harder.

3.5.2 Normalizing the Input

Before the model can be trained and used the speech, now encoded, must be normalized. For speech processing this is typically accomplished by normalizing the mean and variance of the input rows in a sliding window over the input matrix. However, as

the clean speech is not accessible in deployment this option is unavailable. Instead I tried three different methods for normalizing the input speech based.

The first looked only on the elements selected by the IBM and calculated the mean and variance based on these. Predictably this doesn't work since the IBM is not random but heavily biased to select the elements with high energy. The second method calculated normalization constants on the clean speech in training and on the noisy speech in testing. Though this worked, the performance was not satisfactory. The third method I tried computes a global mean and variance over the entire training set and uses these constants both for training and testing. I found this to work the best.

In hindsight this was probably a poor choice. Despite it showing good results on the validation set which is often accepted as representing unseen conditions, and despite this often being true in regards to aspects such as transcript and speaker, it makes no attempt to cover also unseen loudness conditions. Thus, the correct normalization constants on the training and validation data would be much more similar than the correct constant on real data.

3.5.3 Optimizing the Hyper-Parameters

Most machine learning systems have hyper-parameters. These are parameters that are not directly optimized by in the training nor are they fixed by the implementation. in this case examples include the number of hidden units, the learning rate,regularization penalties and, though not strictly part of the model, I will include also context. Conveniently there exists a Kaldi recipe that have fixed all of these to their optimal values for the purpose of speech recognition. However, as the purpose of this system is not to discriminate between tri-phones but rather encode speech pattern with sufficient detail to allow sampling it decide to re-evaluate the optimal value of the number of hidden units. Further, the significance of context is also different. Whereas the amount of additional information gained from more context has quickly diminishing marginal returns when each frame is complete with reliable units, these returns diminish less quickly when additional frames are partially masked in different frequencies. As such, I re-evaluate also the optimal context length.

To find the optimal number of hidden units I fix the context length to 5 and evaluate the reconstruction performance on speech that has been masked with babble noise at -5dB, 5dB and 15dB. The results are presented in figure 3.3. Since the most commonly used hidden layer size for the purpose of speech processing with TIMIT is 2048 units, I was expecting to find an equal or greater number to be optimal for the purpose of sampling. It is thus slightly surprising to find a peak at 1024 units with clear deterioration when increasing the size further. But the result is consistent between SNRs and this is the setting I use throughout the rest of the report. Curiously, looking at the performance of the RBM when acting as an autoencoder the optimal value is more likely 2048. Indeed, this use of the RBM can be considered more similar to pretraining a DNN, slightly reconciling the discrepancy noted earlier.

Similarly, to find the optimal context length I fix the number of hidden units and other

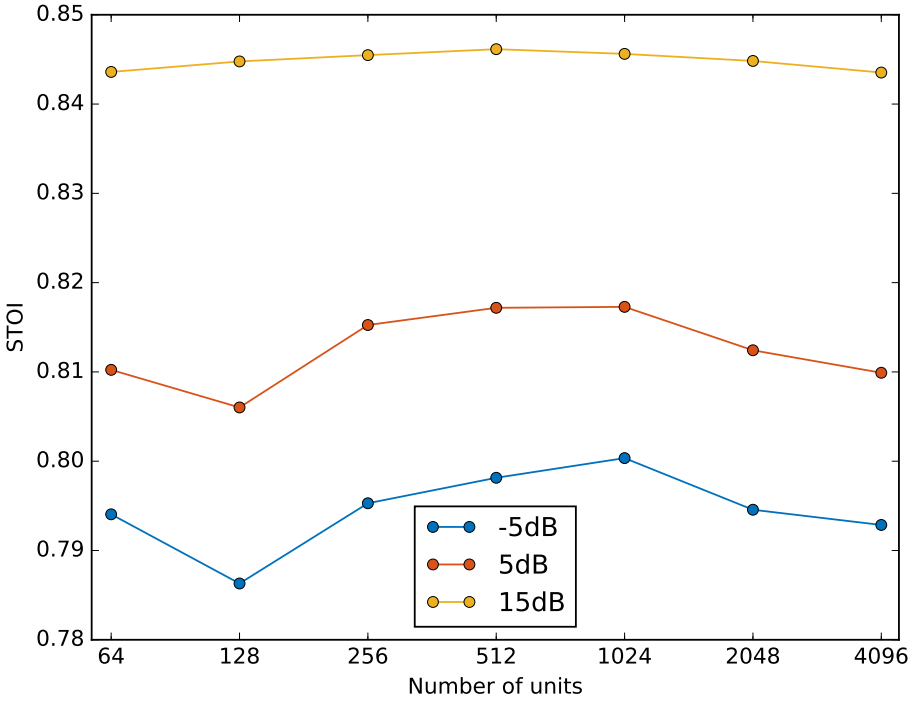


Figure 3.3: An illustration of the effect of model size on performance. It shows the how the quality of reconstructed speech varies with the number of hidden units in the model. It indicates that the optimal value is 512 or 1024.

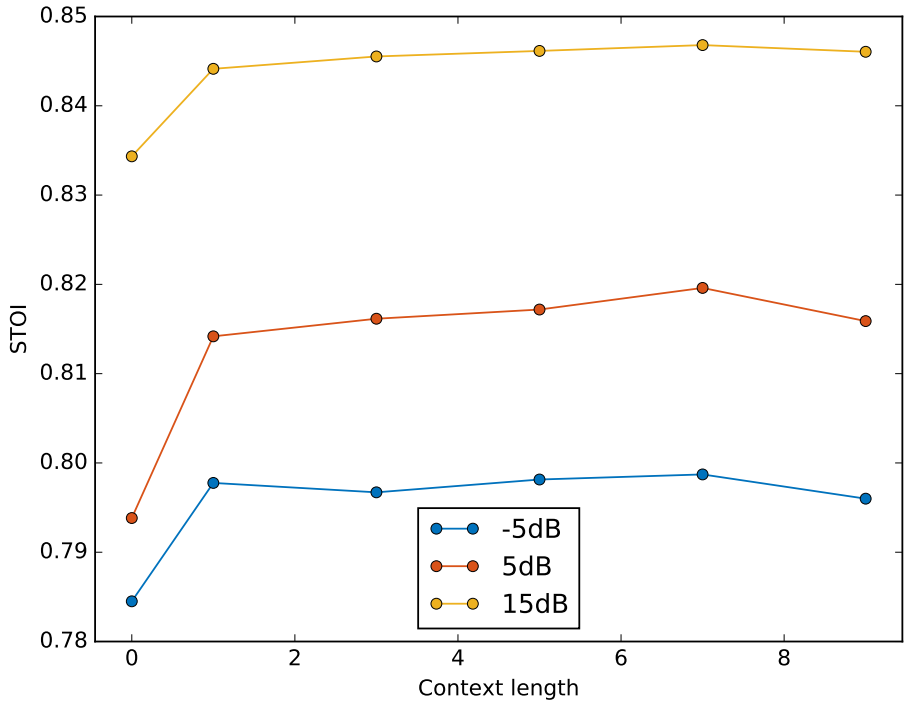


Figure 3.4: An illustration of the effect of input context length on model performance. It shows how the quality of the reconstructed speech varies with the number of adjacent speech frames that are stacked. It indicates that using at least some context is important and that using 7 frames on each side is optimal.

parameters as above, then train one model for each setting and evaluate its reconstruction performance on masks that result from three realistic noise conditions. The results are presented in figure 3.4. Across all noise conditions 7 frames of context on either side of the central frame are found to perform the best. As expected, this is higher than the 5 commonly used in literature, but not much. The performance gained is also not high, with marginal return being very for contexts longer than 1 frame.

3.6 Monophone Extension

Knowing that monaural speech enhancement is a hard task, this project investigates if it is possible to improve performance using an additional source of information such as transcripts of the speech. One scenario in which this is realistic is processing TV-shows where subtitles exist to make the audio more intelligible and accessible for viewers with impaired hearing without the need to read the subtitles. This section describes the modifications made to the reconstruction system in order to make use of phoneme transcripts and additional challenges that arise when training and using the system. Finally it describes how test transcripts are transformed into phone transcripts.

3.6.1 Implementing the Extension

There are at least two ways in which the system presented in section 3.5 can be modified to utilize transcript data. Both require that the transcripts have been translated to phone transcripts and aligned with the speech such that each code vector maps to one phone. This process is discussed in more detail in 3.6.3 and as it has been done previously access to perfect alignment is assumed for the time being.

The first way is to implement categorical units in addition to the existing binomial and Gaussian units. Once implemented the speech vectors would be augmented with a one-hot coding of the corresponding monophone. The other method takes a divide and conquer approach, creating one model for each monophone and then splitting the data at training and testing time to redirect input features labeled with a given monophone to the corresponding model. Advantages of the first include weight sharing and its compactness. Since only a single model is created it benefits from all the training data and only one instance needs to be saved and retrieved. The advantage of the second is being simple, easily testable and avoiding an implementation that is tightly coupled with the existing Kaldi code. Having had problems neatly extending the RBM to perform sampling, I chose simplicity of implementation of the second method.

As can be seen in figure 3.5 the new monophone model achieves lower performance than the previous aggregate model despite addressing what should be an easier problem.

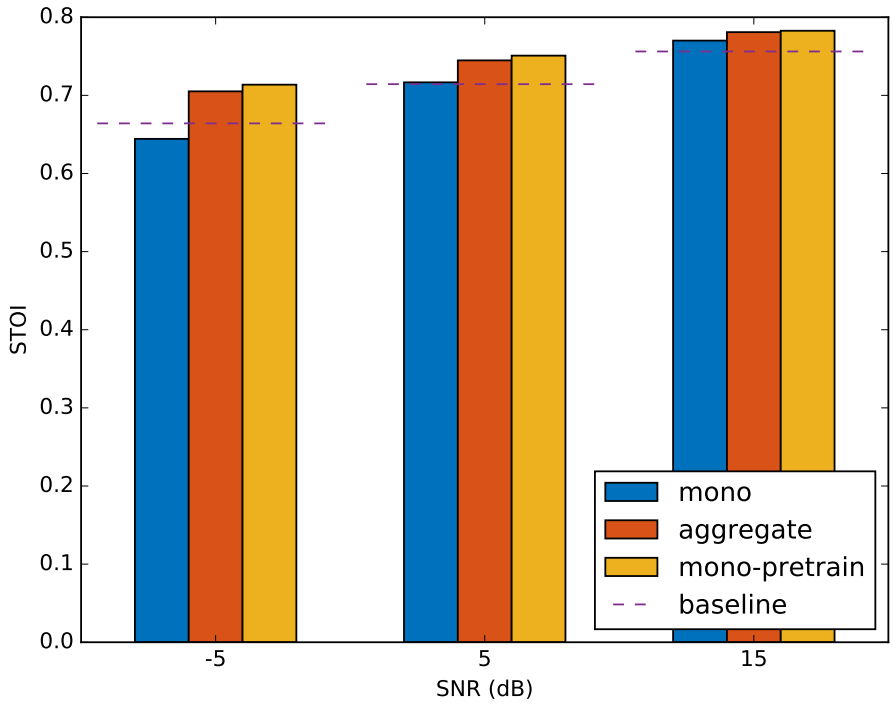


Figure 3.5: A comparison of the intelligibility of speech as reconstructed by three models under three different noise conditions. It shows that using separate models for each monophone can be beneficial, but only if the problem of insufficient data is addressed by pretraining the models. The baseline indicates the intelligibility of speech prior to reconstruction.

3.6.2 Pretraining

One problem with training one model for each monophone that was alluded to earlier is that each model receives less training data. This problem is further aggravated by the fact that all phonemes are not equally common, with my brief analysis showing that the 50% most common monophones account for 94% of all occurrences not counting silence Indeed, monophones with few occurrences tend to be associated with higher error rates. This does not establish causality, perhaps some monophones are acoustically complex and therefore both rarely used and hard to model, but it is a clue to a possible solution.

One way to provide extra training data is to simply use a bigger corpus and I accomplish this by switching from TIMIT to WSJ. However, simply expanding the data set increases the computational cost of training the model and because of the uneven distribution of monophones, the rare ones may still have an insufficient number of occurrences. Since different monophones, despite corresponding to distinct sounds, share structure such as the correlation between the harmonics, it is possible to learn about patterns in one from patterns in another. I thus initialize all monophone models from an already trained aggregate model.

As can be seen in figure 3.5 this clearly improves the performance of the monophone extension, merely surpassing that of the aggregate model. But considering that the transcript holds sufficient information to synthesize an intelligible voice, this is still not satisfactory.

This lack of improvement can be explained in one of two ways, either the extended system is badly designed and unable to utilize the information contributed by the monophones or knowing the monophone of the central speech frame does not contribute much information. Now, speech is a dynamic process and the manifestation of a monophones depends on its context. For this reason, in speech recognition, the atomic phonetic unit is usually not the monophone but the triphone. The implication for the application at hand is that conditioning on the monophone may not sufficiently reduce the subspace of possible speech vectors.

One solution to this problem is to make triphone dependent models instead of monophone dependent. However, whereas there are only 38 monophones there are 14,970 triphones making this approach infeasible. A better solution would thus be to implement the first approach to making a phoneme dependent model mentioned in 3.6.1, but the next section will conclude that this should be left as future work.

3.6.3 Alignment

Transcripts are typically available in text form, but sentences and words are big phonetic units spanning much longer time frames than is practical to model for the purpose of speech enhancement. They are also complex in the sense that they are long sequences of sounds and that many such sequences can be produced. It is much simpler to model the atomic units of speech called phonemes each of which correspond

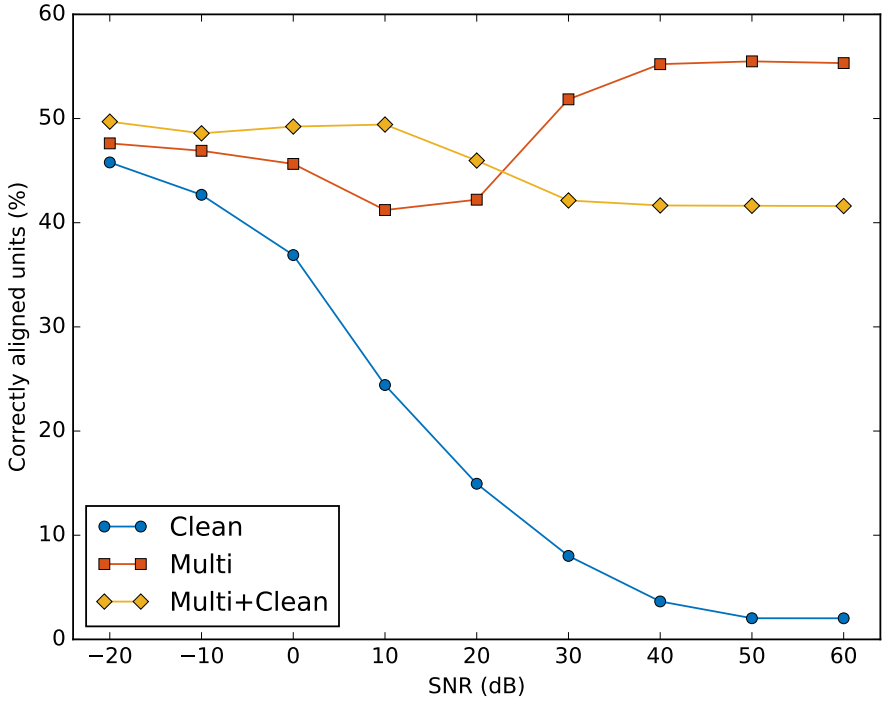


Figure 3.6: An illustration of the ability of the HMM-GMM model to align noisy data depending on what data was used to train it. It shows that this ability is low in the target region of -5dB to 15dB, suggesting that a better technique is needed.

roughly to one of a limited number of sound that humans produce when speaking. It is thus necessary to convert from the text transcripts to time aligned phone transcripts. Conveniently such transcripts are available for the TIMIT corpus but it is not generally recommended to use these for training as the precise alignment may not be consistent with any automatic alignment procedure used at test time.

The primary process by which text transcripts are traditionally transformed into phone transcripts is called forced alignment. It is a well established process in speech recognition and as such routines exist in Kaldi that perform it using Hidden Markov Model (HMM)-GMM models. Here I evaluate how well these methods perform on noisy data. I use the alignments produced by a model trained on clean data and applied to clean data as ground truth. I then compare these alignments to those produced by each candidate model on noisy speech at varying SNR reporting the proportion of frames that are aligned with the wrong monophone. The candidate models are one trained on clean speech, one trained on speech mixed with babble noise at SNRs ranging from 0dB to 20dB and one trained on both the clean speech and the noisy speech.

The results are presented in figure 3.6 and one can see that when no attempt is made to address the noise the performance quickly deteriorates to a point where 20-40% of all frames are assigned the wrong monophone. Remember that typical SNRs that a listener encounters are in the range -5dB to 15dB. It is also clear from the figure that the multi-condition training, while predictably achieving its best performance in the

range of SNRs it had been conditioned for, does not improve the alignment. One potential explanation for this is that the HMM-GMM model is trained in an unsupervised fashion and thus finding clusters that are not in fact the monophones but a coincidental combination of speech and noise. This is probably why state of the art forced alignment models use supervised, discriminative models and I suspect that implementing and training such a model would drastically improve alignment performance.

However, as both this problem and the problem of modeling triphones would both have to be solved in order to obtain a working transcript-dependent system both tasks are left as future work.

3.7 Code Masking

The work done thus far has assumed the existence of the IBM. While doing so simplifies and speeds up both development and evaluation, the absence of a way to estimate the mask renders the presented system unusable in its current form. This section addressed this by implementing a version of the mask prediction component that does not depend on oracle knowledge but rather estimates the mask from the available noisy speech.

3.7.1 Managing Assumptions

Kaldi provides many binaries and scripts to set up and train various models including DNNs. These can usually be glued together with bash scripts, as I have, without paying much attention to their implementation details. However, it is designed solely with speech recognition in mind and so makes assumptions on how these models will be used that are generally compatible with speech processing workloads but not necessary speech enhancement. Relevant examples include assuming that targets are vectors and that targets are one-hot coded categorical data. While the former assumption is easily spotted as it makes it impossible to compile code that tries to use a matrix target and can easily be fixed, the latter does not raise any errors.

Recall that the gradient from the cross entropy error function is $\frac{\partial E}{\partial y} = \frac{y-t}{y(1-y)}$ and that the derivative of the sigmoid layer is $\frac{\partial y}{\partial a} = y(1-y)$. Combined these simplify to just $\frac{\partial E}{\partial a} = y - t$, an optimization often performed to improve numerical stability. Normally the error function and the back propagation are implemented in different components and to implement this optimization one would have to explicitly address the case where the output layer is of a softmax type and the error function is cross entropy. In Kaldi however, as this is assumed to be the only case, the optimization is implemented in the error function. Thus, attempting to use a sigmoid output layer will silently propagate the wrong gradient.

Once identified, this problem is easily rectified by implementing a sigmoid unit to be used only in the output layer. Other problems like it are also relatively easy to fix when

identified but knowing that they exist vastly expands the scope of troubleshooting, slowing down development.

3.7.2 Optimizing the Hyper-Parameters

Despite the RBMs used in 3.5 and 3.6 and DNN used in this section being similar and sharing many types parameters, they are different models addressing different problems. As such the optimal hyper-parameters are likely to be different and need to be optimized separately.

The first such parameter is learning rate of which there are three, one for the first layer of greedy layerwise generative pretraining, one for pretraining of subsequent layers and one for discriminative fine tuning. Since generative pretraining is essentially the same process for my speech enhancement as for a speech recognition system, optimal values found by others and presented in Kaldi recipes are used for the two first. For the learning rate in fine tuning the fact that I use a sigmoid output layer and 704 means I should use a learning rate that is about an order of magnitude smaller. As this is a rough estimate, I conduct an experiment to find the actual optimal value. Starting at the given learning rate of 0.008 I train the network and halve the learning rate until I'm convinced the performance start to decrease. I find the for the two first setting the training does not converge, then from 0.002 performance improves until the optimal rate of 0.000125 before decreasing for at least an additional two steps.

The second parameter I optimize is the depth of the network. Masking performance improve slightly from 1 layer to two then starts deteriorating.

Lastly I optimize the number of hidden units in each layer, I present these results in figure. I do not experiment with layers of different size. Starting at 256 units per layer I double the size until performance plateaus. Although diminishing marginal returns are clearly seen, at 2048 units performance is still increasing. At this point however, only fine tuning the network takes over 204 hours running on the CPU and despite much effort I was unable to make it run reliable utilizing the CPU. Considering that during optimization I only address the problem of matching noise conditions and that addressing multiple, mismatching conditions benefits from orders of magnitude more data, I choose 1024 as the layer width for the final evaluation of the system.

Optimizing parameters independently as I have done does not necessarily yield the best solution. For instance, the optimal number of hidden units per layer is likely dependent on the number of layers as the total number of parameters scale with the product of the two. And though the substantial computational cost of performing grid search over multiple parameters jointly can be mitigated by randomization, treating hyper-parameters as independent is simple and should yield a sufficient optimization.

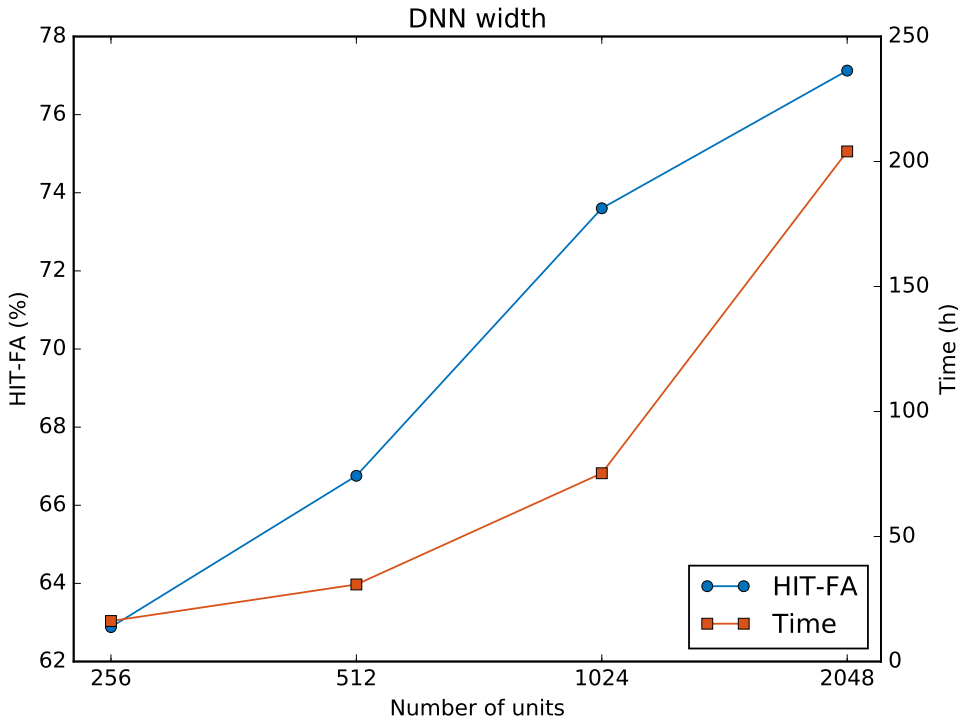


Figure 3.7: An illustration of the effect of model network on performance. It shows how the computation needed increases faster than the performance, and it must be decided when the marginal improvement in performance is not worth the marginal increase in computational cost.

Chapter 4

Evaluation

To establish the usefulness of the systems implemented, this chapter describes experiments designed to evaluate this as well as the results and implications. It also describes experiments that explore the limits of the conclusions drawn from earlier experiments. The systems are implemented in the same order that they were implemented and not the order in which they would execute in a deployment.

4.1 Reconstruction

The first subsystem evaluated is the reconstruction system. It does not exist in isolation however, but depends on all other components though it should be noted that an oracle is used in place of a real mask prediction system. As such, it's performance is the product not only of itself but also of the method by which data is prepared and evaluated as well as the speech coding. This section describes three experiments designed to evaluate what performance can be expected, to investigate assumptions and limits of that performance and to identify potential areas of improvement.

4.1.1 Novel Conditions

While implementing the reconstruction subsystem evaluations were performed using only masks from a single source of noise. Other noise may result in masks with different properties and different effect on the performance and it desirable to know how much this may vary. Further, since all hyper-parameters were set in respect to masks by this one noise, the resulting model may be overfitted.

To investigate how well the model performs in general masks are computed for four different noises at three different SNRs, the masked speech reconstructed and its objective ineligibility estimate computed. The results are presented in figure 4.1 where 'cross' indicates crosstalk from another speaker, 'fact' indicates factory noise, 'pub' for pub is a different recording of babble noise and 'rest' for restaurant is the same recording of babble noise as was used in development.

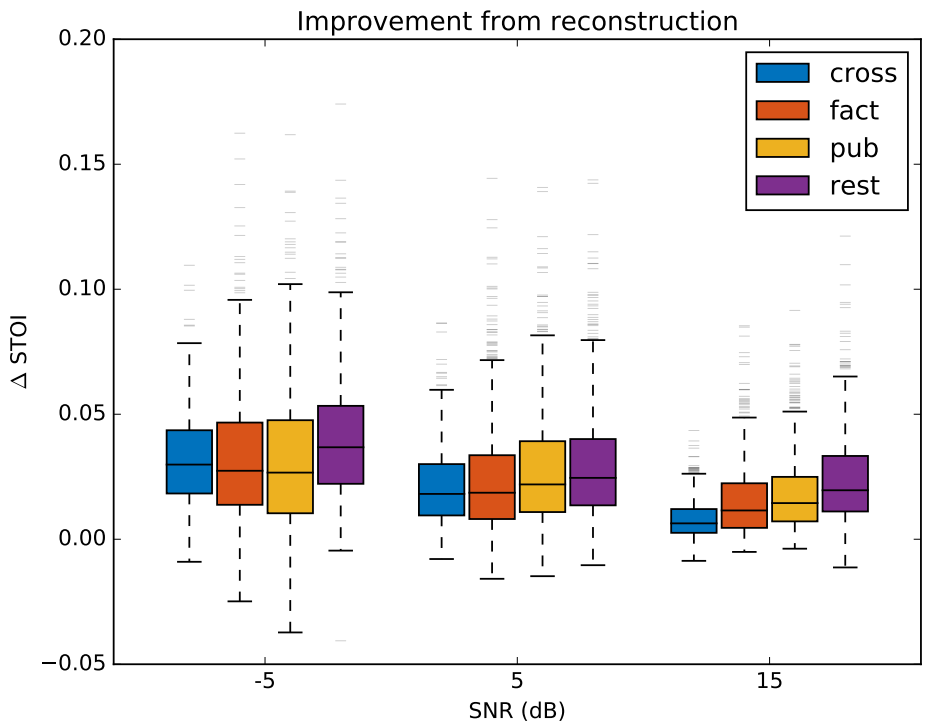


Figure 4.1: An illustration of how much reconstruction improves intelligibility across a range of conditions. The conditions span three SNRs and noises from four sources, another speaker, a factory, a pub and a restaurant. It shows that for the typical utterances quality is improved, but not substantially. The superior performance in the restaurant noise used in development suggests the model was overfitted.

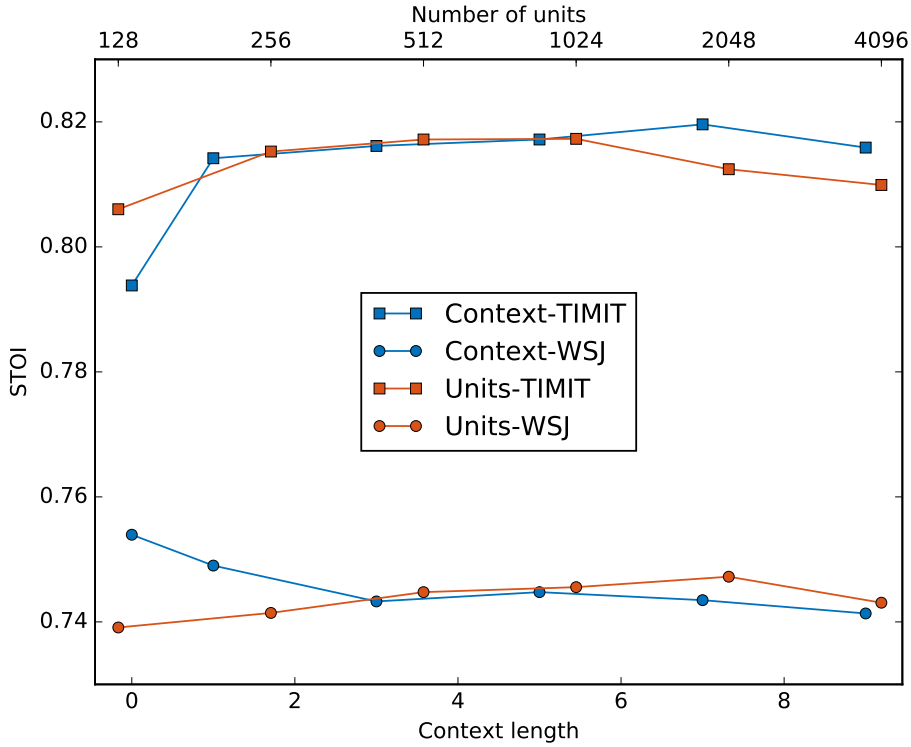


Figure 4.2: An illustration of how the speech used influences both the optimal parameters of a model and the measured speech intelligibility. It shows how speech from TIMIT is consistently considered more intelligible than speech from WSJ that has undergone identical processing. It further shows how the optimal number of hidden units is 2-4 times larger for WSJ but the optimal context is smaller. Note that blue and red lines correspond to context and hidden units respectively while squares and circles indicate TIMIT and WSJ respectively.

From the figure we can tell that the typical improvement under training conditions is consistently better than under novel conditions suggesting that the model may be overfitted to this condition. Though consistent, the difference isn't substantial especially considering the relatively high standard deviation. Indeed there are many instances in which the model performs worse under training conditions than the novel conditions, especially when the noise is mild (15dB). On the other hand this may be a coincidence due simply to that type of noise being easier to deal with and it would be interesting to investigate if optimizing the model for another noise would reverse this pattern.

Another observation that can be made from the figure is that while the model typically achieves improvements it will sometimes also deteriorate the speech. Though this is a relatively rare occurrence and the deterioration is mostly small, the fact that it does occasionally happen may deter one from deploying such a subsystem.

4.1.2 Corpora

When developing a machine learning system one typically divides the set of all data into training, validation and test sets to ensure that the reported performance is not overconfident and will reflect performance on new data. The generalization from one such set to another was discussed in the previous section. However, this approach still assumes that the set of all collected data is representative for the type of data the system is intended to handle.

Figure 4.2 illustrates two issues that arise if one is not careful. First, we notice how the objective ineligibility metric is consistently 0.06 higher for TIMIT than for WSJ. While this does not render results achieved with either corpus invalid, it does mean that attempts to compare performance of different systems developed on different corpora is going to be very hard. Second, we notice that the optimal hyper-parameters vary between the corpora. For WSJ the optimal context length is 0 while for TIMIT it is 7. Similarly, for TIMIT the optimal number of hidden units peaks between 512 and 1024 after which it start deteriorating while for WSJ it only starts deteriorating after 2048 units. This suggests that a model that has been found by one researcher to be optimal may not be optimal to another even if they both target the same underlying data but by proxy of different corpora.

Though hardly surprising, this discrepancy is worthwhile keeping in mind whenever making comparisons between systems that are not developed in identical conditions.

4.1.3 Phase and Amplitude

When speech is decoded the real spectrum matrix is elementwise multiplied with the phase spectrum matrix to produce the complex spectrum that can then be processed with the Inverse Fast Fourier Transform (IFFT) to produce time domain signals. Thus, by applying a binary mask to the power spectrum the phase spectrum is also masked and reconstructing these masked values will undo this masking. Now, if the SNR in the phase spectrum is the same as in the power spectrum, noise has been added back in. To control for this, in experiments up to this point, the clean phase has been used for decoding. Conversely if the phase is clean, reconstructing masked units may add information back into the signal was previously unavailable. Figure 4.3 compares shows the different outcomes of these two approaches in three different noise conditions. We can see that whenever the clean phase is used the speech improves on average but when the dirty phase is used it deteriorates. Consequently, the results presented in the section 4.1.1 are not obtainable unless the phase is also enhanced. One interpretation of this is that the reconstruction is bad since noise the phase is largely imperceptible if the SNR in the amplitude is sufficiently high [13]. On the other hand, it may be easier to improve the SNR in the phase than to further increase the SNR in the amplitude, making enhancing also the phase one interesting topic of future research.

Another potential flaw in the reconstruction system is the assumption that the masked amplitude has a sufficiently high SNR that it can be approximated by clean speech. The LC value used to compute the IBM bounds this SNR from below at -6dB and the

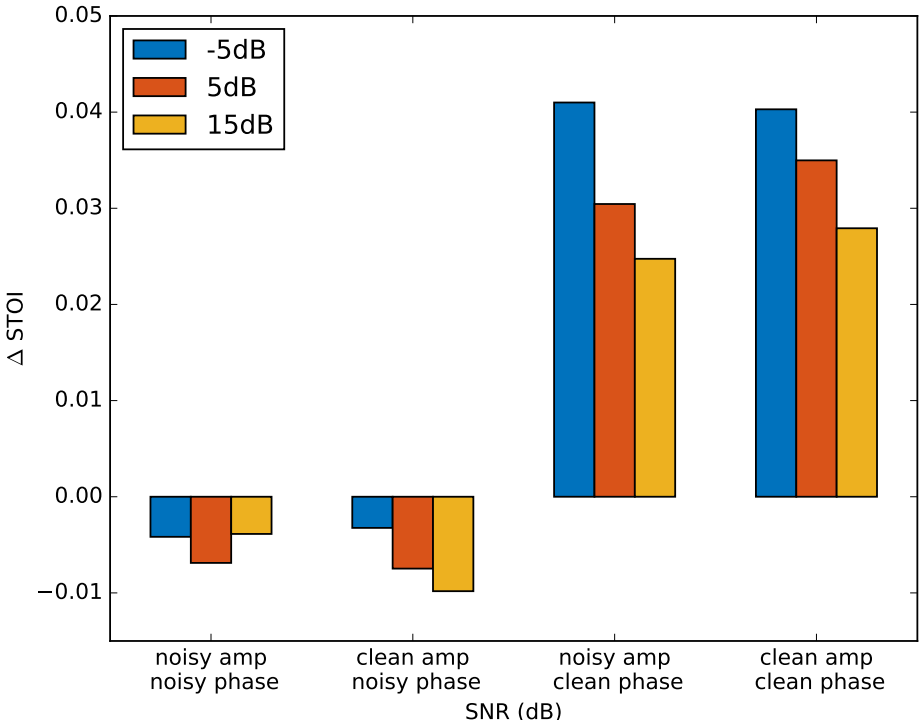


Figure 4.3: An illustration of the effect of using dirty and clean data on reconstruction performance across three different conditions. It shows how reconstructing masked units will have a positive effect when the phase is clean but a negative effect when it is dirty, suggesting that there's reason to enhance also the phase. It further shows that the effect of noise in the amplitudes masked as clean, is noticeable but small, suggesting that it may be beneficial to optimize the Local Criterion (LC).

effect of this is also shown in figure 4.3. When the noisy phase is used for decoding, the results are inconsistent but on average the performance is slightly worse when the clean amplitude is used. When on the other hand the phase is clean, the clean amplitude consistently yields better performance. The effect is not substantial and the improvement from optimizing the LC would likely be lower as it would result in a less dense mask. As such I don't investigate the optimal setting of the LC, I just note that it has an effect and should be optimized before deployment.

4.2 Masking

The second subsystem developed developed estimates and applies the IBM to enhance noisy speech. This section first evaluates how well the system generalizes to new conditions, then investigates it ability to isolate a known speaker among other speakers and lastly discusses the merits of the metrics used to compare systems and performance.

4.2.1 Generalization

To evaluate realistically the performance of the masking system it is trained on speech mixed with four noises at SNRs from 0dB to 10 dB in steps of 1dB. In order to maximize variety within the dataset each noise recording is used to produce 60 shorter samples, each 30s long and offset from the previous by 1s. To also restrict the size and consequently the training time 1 in 4800 mixture samples is actually produced and used for training. The noise recordings used are background noise from a bar, a car, a factory and a restaurant. Because the reconstruction subsystem was found not to improve intelligibility on average when the phase is noisy, it is not used here. Instead the soft mask produced by the DNN is used directly has this has been found to improve speech quality over using the binarized version [26].

Once trained, the system is evaluated on novel speech mixed with restaurant noise and pub noise representing seen and unseen babble noise respectively. The intelligibility of the speech before and after masking is evaluated using the STOI metric and the improvement achieved by applying the mask is presented in figure 4.4.

The figure shows the mean improvement achieved at various SNRs for the two each of the two sources of noise. It reveals that under most conditions the mask does not improve ineligibility, with the exception being pub noise at 5dB and 10dB. This is clearly inferior to other published results, one of which report improvements of 0.11 in babble noise at 0dB [26]. Considering that each noise condition that the model addresses is trained with only a fraction of the data previously used to train a model for a single condition, one possible way to address this issue is simply using larger quantities and more varied data.

Further, the intelligibility gains presented in 4.4 compares speech that has been encoded and masked to speech that has only been encoded. When the same comparison is made between speech that is encoded and masked and the unprocessed mixture, the

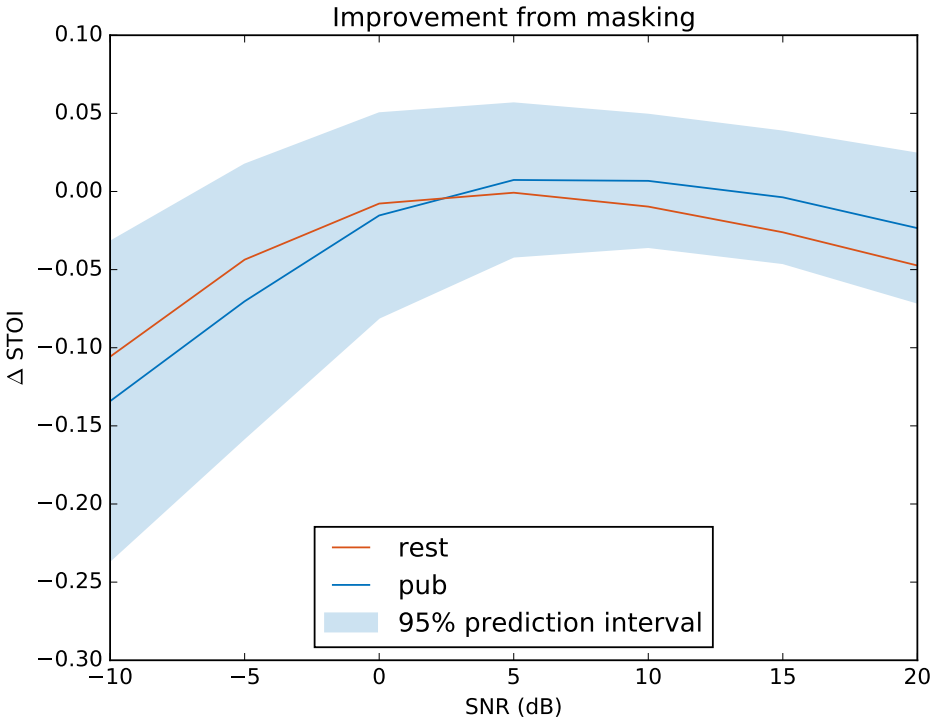


Figure 4.4: An illustration of how much masking improves speech intelligibility across a range of conditions under seen restaurant noise and unseen pub noise. It shows that the masking typically don't improve intelligibility, suggesting masks are estimated poorly. That the masking improves intelligibility more under pub noise suggests either that the system generalizes well or that this noise is inherently easier to handle.

SNR (dB)	-10	-5	0	5	10	15	20
Rest	0.396	0.477	0.554	0.606	0.626	0.622	0.607
Pub	0.200	0.285	0.380	0.474	0.542	0.549	0.540

Table 4.1: A comparison of the masking performance under a range of seen and unseen conditions. The restaurant noise used in training represents seen conditions and its superior performance suggest the system does not generalize well and is perhaps overfitted.

outcome is consistently that masking deteriorates intelligibility. I therefore suggest that using a coding method that has minimal impact on the speech quality and intelligibility is important and a logical improvement.

Knowing that the training conditions all have an SNR of 0dB to 10dB, it is not surprising to find that the performance under the known, restaurant noise peaks in this interval. More surprising is the fact that the system performs better under unknown, pub noise suggesting that this particular noise may be easier to deal with. To validate this I compare the performance of the system also in terms of HIT-FA rate. This comparison is presented in table 4.1.

In the table, contrary to what the STOI scores suggest, it appears the system performs substantially better under the known, restaurant condition. This suggesting that it may be overfitting to the known conditions and lends support to the opinion that using more training data would benefit performance.

However, it performs much worse than when it was trained to on a single noise condition as in the optimization experiments in section 3.7.2 where an average HIT-FA rate of 0.736 was achieved. But both rates are inferior to the 0.81 reported for DNNs in other studies [26].

4.2.2 Whitelisting and Blacklisting

One of the original goals of this project was to be able to filter out a single voice. While much emphasis has changed from this to the ability to remove background noise, I revisit the goal briefly in this section.

I envision two scenarios each with one known speaker. In the first, similarly to whitelisting, the known speaker is the target and all other speech should be attenuated. In the second, similarly to blacklisting, all speakers except for the known speaker is the target and only the known speaker should be attenuated.

To evaluate the performance in these scenarios I train and evaluate two systems. One where speech from a single speaker is used as signal and speech from 9 other speakers is used as noise is evaluated on unseen utterances by the target speaker masked by a 11th speaker. Another where speech from the 9 speakers are used as signal and speech from the single speaker is used as noise is evaluated on utterances from the 11th speaker masked by unseen utterances from the known speaker. In both cases the

SNR (dB)	5
Whitelisting	0.513
Blacklisting	0.552

Table 4.2: A comparison of the ability of the system to isolate a known speaker and its ability to remove a known speaker. The result suggests that it is easier to attenuate a known speaker than it is to amplify them. The performance is similar to that achieved in general conditions, being slightly better than unseen noise but slightly worse than seen noise.

SNR is fixed at 5dB resulting in crosstalk that subjectively makes the sentences hard to follow. The HIT-FA rates are presented in table 4.2.

Comparing the results to those obtained in the general case in the previous section, both blacklisting and whitelisting achieves performance slightly better than the novel case presented there thus indicating little or no gain in ineligibility. I find that this agrees with my subjective experience and I judge the intelligibility to be about equivalent. However, I find the masked speech is less distracting.

4.2.3 Metrics

Throughout the development of the masking subsystems three different metrics have been used. First, cross-entropy, which is a proxy for accuracy, is used as error function in the training. Second, the HIT-FA rate is used to compare and optimize hyperparameter configurations as it is easy to compute and have been to correlate well with ineligibility. Third, STOI is used in the evaluation as it is independent of this specific enhancement methods and also correlates strongly with ineligibility. All are used as a way to estimate how intelligible the resulting speech will be.

However, as can be seen in figure 4.5 these metrics will not always agree on how intelligible a sample is. Indeed it is not hard to pick two samples such that two metrics will substantially disagree. However, when two samples belong to the same series or the results of the same noise, STOI and HIT-FA will agree when one is substantially better than the other. The accuracy metric on the other hand can have very low correlation with another metric within a series, as indicated by the nearly vertical and horizontal lines.

The significance of this is twofold. Firstly, as with the corpora, it suggests that a system that was wound superior to another by one metric, may by a second metric be inferior. Secondly, perhaps more important, it reaffirms the importance of optimizing for the most meaningful metric. In this case, this would be STOI, but there is no known way of directly optimizing it in machine learning models such as DNNs. The second choice is HIT-FA which maximizes AI which in turn correlates positively with speech intelligibility. This is possible and has been shown to yield improvements over optimizing cross entropy. As such it would be a good next step in improving the performance of this masking system.

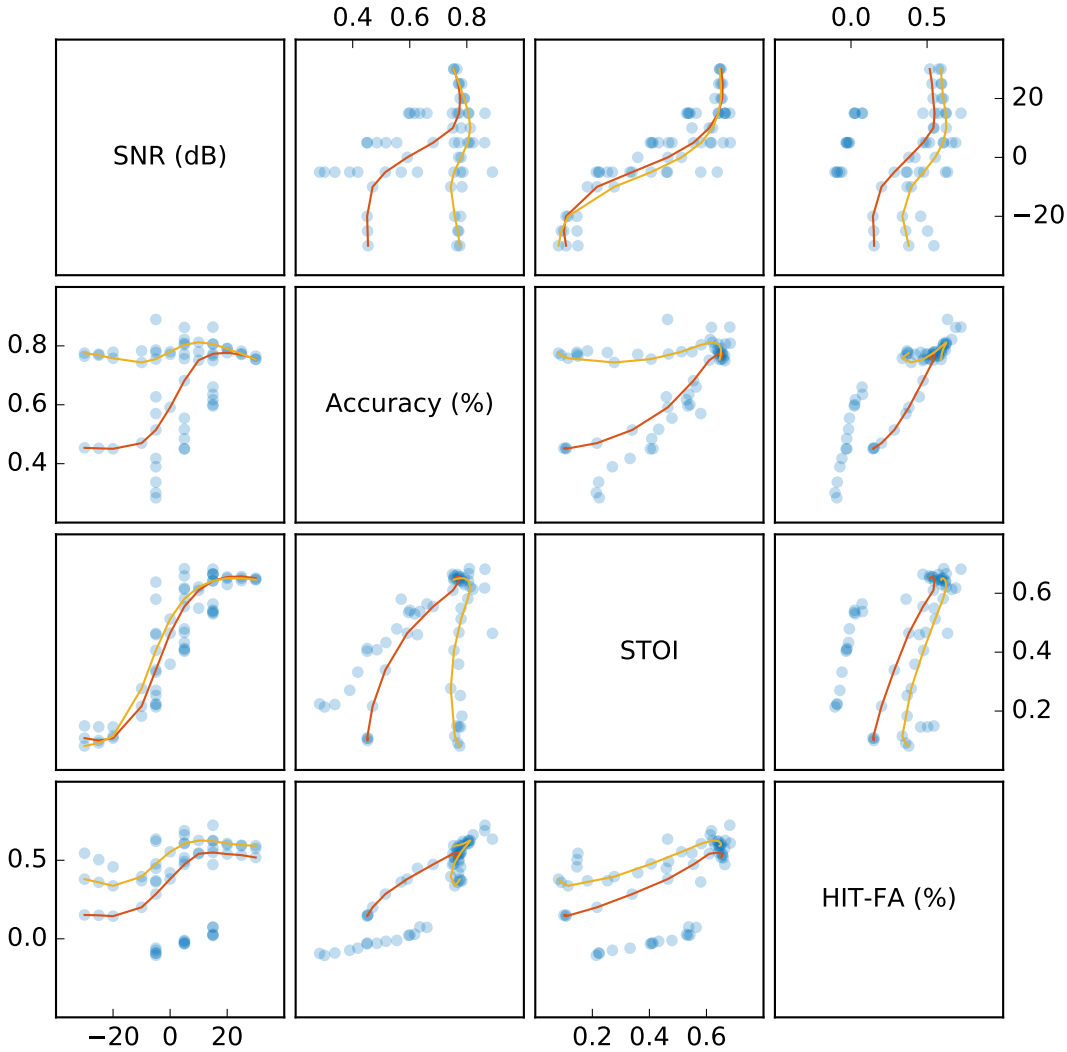


Figure 4.5: An illustration of the pairwise correlation of different the metrics used to determine the performance of the presented system. It shows how STOI and HIT-FA can disagree on what utterance is more intelligible but that they will agree if the utterances were corrupted with the same noise. It further shows that accuracy does not correlate well with either STOI or HIT-FA suggesting that cross entropy is a bad error function.

Chapter 5

Conclusion

This report has described the process of implementing and evaluating a speech enhancement system that draws inspiration from recent progress of graphical generative models in computer vision and the IBM framework from Computational Auditory Scene Analysis (CASA). In section 3.7 a discriminative masking system that, given nosy speech, estimates a mask that removes noisy elements is implemented, trained and optimized. It is evaluated in section 4.2.1 which concludes that the due to aggressive speech coding, no improvements are achieved. In a subjective evaluation in section 4.2.2 I find that despite being unable to improve intelligibility, when interference is crosstalk and one of the speakers are known, it is able to make it less distracting. Similarly, in section 3.5 a generative reconstruction system that, given fragments of speech, reconstruct the missing pieces is implemented, optimized and trained. It is evaluated in section 4.1.1 to find that a small gain in intelligibility is possible but section 4.1.3 shows how this is contingent on the phase containing no noise. Finally, section 4.1.2 and 4.2.3 argues that any conclusions drawn about the performance of a system on one set of data or using one method of evaluation will often not to to be true for another set of data or another method of evaluation.

With quality deterioration from speech coding identified as a key weakness in the presented system, this presents one evident opportunity for improvement. Modern DNN have a record of being able to deal with large inputs and outputs and simply using the FFT output would be one easy way in which to address this. Where computational expense is an issue, further research into speech coding techniques for compression and transformation could reveal a higher quality, lower dimensional coding that is amenable to masking or direct mapping.

Another opportunity for improvement is identified in section 3.6 which argues that, because monophones vary a lot depending on their context, the more stable triphone should be used to annotate frames. Having also pointed to the inability of flat start, maximum-likelihood clustering to handle, this improvement would have to be coupled with better initialization and training of the clustering model or better a discriminative model.

In section 4.2.3 the poor correlation between accuracy in mask prediction and resulting

speech intelligibility is alluded to. While this is not a new realization and HIT-FA was introduced as a superior option in 2.2.1, the presented system would greatly benefit from its adaption.

The last opportunity for improvement is using a faster and more scalable platform. In section 3.7.2 high computational cost prohibits the use of a more flexible model and again in section 4.4 high computational cost prohibits the use of a larger data set which it is argued would improve the systems ability to generalize to new environments.

Bibliography

[1] Hearing aids - Live Well - NHS Choices.

[2] Douglas S Brungart, Peter S Chang, Brian D Simpson, and DeLiang Wang. Isolating the energetic component of speech-on-speech masking with ideal time-frequency segregation. *The Journal of the Acoustical Society of America*, 120(6):4007–4018, 2006.

[3] Geoffrey A Fowler. Hearing Aid? No, Soundhawk Is an Ear Wearable.

[4] Kun Han, Yuxuan Wang, and D Wang. Learning spectral mapping for speech dereverberation. *Proc. ICASSP, to appear*, 23(6):982–992, 2014.

[5] O. Hazrati and P. C. Loizou. Tackling the Combined Effects of Reverberation and Masking Noise Using Ideal Channel Selection. *Journal of Speech, Language, and Hearing Research*, 55(2):500–510, 2012.

[6] Yi Hu and Philipos C. Loizou. Subjective comparison and evaluation of speech enhancement algorithms. *Speech Communication*, 49(7-8):588–601, 2007.

[7] Yi Hu and Philipos C Loizou. A new sound coding strategy for suppressing noise in cochlear implants. *The Journal of the Acoustical Society of America*, 124(1):498–509, 2008.

[8] Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis. Deep learning for monaural speech separation. *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 1581–1585, 2014.

[9] Gibak Kim, Yang Lu, Yi Hu, and Philipos C Loizou. An algorithm that improves speech intelligibility in noise for normal-hearing listeners. *The Journal of the Acoustical Society of America*, 126(3):1486–1494, 2009.

[10] Karl D. Kryter. Validation of the Articulation Index. 34(11):1698–1702, 1962.

[11] Nicholas D Lane, Petko Georgiev, and Lorena Qendro. DeepEar: Robust Smartphone Audio Sensing in Unconstrained Acoustic Environments using Deep Learning. *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '15*, pages 283–294, 2015.

[12] Ning Li and Philipos C. Loizou. Effect of spectral resolution on the intelligibility of ideal binary masked speech. *The Journal of the Acoustical Society of America*, 123(4):EL59–EL64, 2008.

[13] Philipos C Loizou. *Speech Enhancement: Theory and Practice*. CRC press, 2013.

[14] Philipos C. Loizou and Gibak Kim. Reasons why current speech-enhancement algorithms do not improve speech intelligibility and suggested solutions. *IEEE Transactions on Audio, Speech and Language Processing*, 19(1):47–56, 2011.

[15] Xugang Lu, Yu Tsao, Shigeki Matsuda, and Chiori Hori. Speech Enhancement Based on Deep Denoising Autoencoder. *Interspeech*, (August):436–440, 2013.

[16] Tobias May and Torsten Dau. Requirements for the evaluation of computational speech segregation systems. *The Journal of the Acoustical Society of America*, 136(6):EL398–EL404, 2014.

[17] P Mowlaee, R Saeidi, M G Christensen, R Martin, and Media Technology. Subjective and Objective Quality Assessment of Single-Channel Speech Separation Algorithms . pages 69–72, 2012.

[18] a Narayanan and Dl Wang. Ideal Ratio Mask Estimation Using Deep Neural Networks for Robust Speech Recognition. *. . . , Speech and Signal Processing (* . . . , pages 7092–7096, 2013.

[19] Arun Narayanan, Student Member, and Deliang Wang. Improving Robustness of Deep Neural Network Acoustic Models via Speech Separation and Joint Adaptive Training. 23(1):92–101, 2015.

[20] Tim Jurgens Ray Meddis Nicholas R. Clark Wendy Lecluyse. BioAid: The Biologically Inspired Hearing Aid, 2012.

[21] Todd A Ricketts. Digital Hearing Aids: Current "State-of-the-Art".

[22] Andrew J. R. Simpson. Probabilistic Binary-Mask Cocktail-Party Source Separation in a Convolutional Deep Neural Network. mar 2015.

[23] Andrew J. R. Simpson, Gerard Roma, and Mark D. Plumbley. Deep Karaoke: Extracting Vocals from Musical Mixtures Using a Convolutional Deep Neural Network. apr 2015.

[24] Deliang Wang. Deep Ensemble Learning for Monaural Speech Separation. 2015.

[25] DeLiang Wang, Ulrik Kjems, Michael S Pedersen, Jesper B Boldt, and Thomas Lunner. Speech intelligibility in background noise with ideal binary time-frequency masking. *The Journal of the Acoustical Society of America*, 125(4):2336–2347, 2009.

[26] Yuxuan Wang. *Supervised Speech Separation Using Deep Neural Networks*. PhD thesis, 2015.

[27] Yuxuan Wang and Deliang Wang. A Deep Neural Network for Time-Domain Signal Reconstruction. pages 6148–6152, 2014.

[28] Mitchel Weintraub. A theory and computational model of auditory monaural sound separation. *Processing*, (August), 1985.

[29] Yong Xu, Jun Du, Li-Rong Dai, and Chin-Hui Lee. An Experimental Study on Speech Enhancement Based on Deep Neural Networks. *IEEE Signal Processing Letters*, 21(1):65–68, 2014.