Graz University of Technology

Master Thesis

Speech Enhancement with Sum-Product Networks

conducted at the
Signal Processing and Speech Communications Laboratory
Graz University of Technology, Austria

by
Georg Kapeller, 0313421

Supervisors:
Franz Pernkopf
Matthias Zöhrer
Robert Peharz

Assessor:
Franz Pernkopf

Graz, November 6, 2014

Acknowledgements

Ich möchte an dieser Stelle 3 Personen aufrichtig danken, ohne die diese Arbeit weder entstanden, noch zeitgerecht fertig geworden wäre:

Franz Pernkopf für seine richtungsweisenden Ratschläge und die außerordentliche zeitliche Flexibilität, die er sowohl während der ersten Jahre als auch in den letzten Stunden dieser Arbeit hatte,

Matthias Zöhrer für seine enormen Anstrengungen meiner Arbeit und Sprache Struktur zu verleihen und

Robert Peharz für seine unverzichtbare und versierte fachliche Unterstützung, ohne die Summen und Produkte weit weniger unbegreifbar geblieben wären.

Abstract

Speech enhancement (SE), as a variant of single channel source separation is an inherently hard problem. Nevertheless, recent research indicates that source-adapted model-based approaches can improve the quality of enhanced speech even in highly adverse SE environments.

In this work we apply Sum-Product networks (SPNs) to the problem of SE by training different source-models and using most probable explanation (MPE) inference to infer the underlying sources from the mixture. To demonstrate the viability of our approach we conducted 2 experiments: The first experiment directly infers the clean speech from a noisy mixture, relying on a separate speech probability estimate. The second experiment infers two separate estimates for speech and noise and combines them by means of a softmask. We evaluate our models on the data of the 2nd CHiME speech separation challenge and report PESQ and PEASS scores for a speaker-dependent and a speaker-independent task, as well as for clean and reverberant speech data.

Our results are in line with other deep learning methods such as RBMs (Restricted Boltzmann machines) or AEs (Auto-encoders) and we outperform non-model based approaches by a large margin. Furthermore we can show that directly estimating the clean speech is less beneficial than independently inferring the underlying sources and combining them via a mask. In our experiment softmasks yield significantly higher scores than binary masks.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

date	(signature)

Contents

List of Tables

List of Figures

List of abbreviations

ABE	Artificial bandwith extension
APS	Artefacts-related perceptual score
ASR	Automatic speech recognition
BM	Binary mask
CDF	Cumulative distribution function
cMPE	Constraint MPE
DAG	Directed acyclic graph
DM	Direct mask
DT	Direct target
FHMM	Factorial Hidden Markov Model
GMM	Gaussian mixture model
HMM	Hidden Markov Model
IBM	Ideal binary mask
IM	Indirect mask
IMCRA	Improved minima controlled recursive averaging
IPS	Interference-related perceptual score
ISM	Ideal softmask
MAP	Maximum apoeriori
ML	Maximum likelihood
MPE	Most probable explanation
NP	Network polynomial
OPS	Overall perceptual score
PD	Poon&Domingos
PDF	Probability distribution function
RBM	Restricted Bolzmann machine
RV	Random variable
SCSS	Single channel source separation
SDc	Speaker-dependent - clean speech
SDr	Speaker-dependent - reverberant speech
SE	Speech enhancement
SIc	Speaker-independent - clean speech
SIr	Speaker-independent - reverberant speech
SM	Softmask
SNR	Signal-to-noise ratio
SPN	Sum-product network
SPNHMM	SPN in conjunction with an HMM
SSA	Spectral-substraction algorithm
TF-bin	Time-frequency bin
TPS	Target-related perceptual score

1 Introduction

Speech Enhancement (SE) is the task of using audio signal processing techniques to improve certain perceptual aspects of speech. Although this broad definition of SE also includes e.g. dereverberation or multi-channel scenarios, the main focus of this work lies on the ill-posed problem of enhancing speech corrupted by additive and uncorrelated noise in one microphone setups.

For most applications this comes down to applying some kind of noise suppression algorithm to a given audio-stream with the goal of enhancing the perceptual quality and intelligibility of speech. Although there is an intuitive relation between these two measures, they describe two different aspects: *Perceptual quality* is a subjective measure that captures *how* the speech sounds. That is of high importance as e.g. low quality speech usually results in an exhausting listening experience and can rapidly fatigue the listener. *Intelligibility* measures the correct phonetic identification, i.e. whether it can be understood *what* was said.

In SE perceptual quality and intelligibility are usually decreased because of some sort of noise. However, noise in this context does not refer to noise types such as white noise but means any unwanted, additive interference with the target speech signal. Typically the noise types used in SE stem from real world situations where the need of SE arises. The most prominent situation is telecommunication, where the transmitted speech is disturbed by background noise, e.g. car noise while driving or babble noise in a restaurant. Another application is as a preprocessor for Automatic Speech Recognition (ASR) systems, as these systems are usually not able to cope with the wide range of possible background noises and suffer a severe decrease in word recognition rates. Yet another domain is the implementation of speech enhancement systems into hearing aids as hearing-impaired listeners especially suffer from the amplification of noisy/low-quality speech.

One would intuitively expect that an improvement in perceptual quality would also yield some improvement in intelligibility and vice versa. However, in speech enhancement we currently face two main challenges: The first is significantly improving the intelligibility of enhanced speech, which resulted to be a much harder task than improving perceptual quality. In the past 40 years many of the algorithms developed achieved substantial gains in perceptual quality but without significantly improving the intelligibility. The second challenge is to minimize the trade-off between quality and intelligibility. Because of various reasons speech enhancement algorithms can produce acoustic artefacts (e.g. musical noise) which do not necessarily affect a possible gain in intelligibility but sound very unnatural to a human listener and thus result in significantly lower perceptual quality [1, Ch. 1].

Much important research has also been done in the closely related field of Single Channel Source

Separation (SCSS) [2–6]. In SCSS the mixture signal is composed by two or more target signals, which ought to be recovered separately. This is usually stated as a problem of *unmixing* the target components. On the contrary, in SE the focus lies on recovering exactly one target signal out of a mixture of possibly many noise sources, i.e. it is stated as *supressing* an unknown number of non-target sources. Furthermore, SCSS is often used to separate two competing speakers, making explicit use of the sparse nature of speech [7]. This target sparsity assumption does not hold for many types of noise, thus some SCSS algorithms may not be directly applicable to the SE domain. Recent research however can be equivalently applied in the SCSS and SE context and therefore we will make no further formal distinction between the two fields.

1.1 Objective and scope

The aim of this work is to develop and evaluate a new type of model driven speech enhancement algorithm. Motivated by the good results of Sum-Product networks (SPNs) on the problem of Artificial Bandwidth Extension (ABE) [8], our idea was to take this deep-learning approach one step further and use it for speech enhancement in a similar fashion. As shown in [9], the field of representation learning, which is a sub-category of deep learning, is currently governed by 4 main models: Restricted Boltzmann Machines (RBMs), Auto-Encoders (AEs), General Stochastic Networks (GSNs) and SPNs. A recent study [10] successfully applied the first three models on the task of SE/SCSS – this work thus completes the evaluation of deep models by evaluating SPNs on the same task.

1.2 Outline

This work is structured as follows: In Chapter 2 we give a formal introduction of the main algorithms and concepts developed in the field of SE and review the relevant literature. Chapter 3 introduces Sum-Product Networks and lays the foundations for Chapter 4, where we present the two main SE experiments we conducted and give an overall comparison of achieved scores. Furthermore, we put the models and results into a broader perspective. Chapter 5 discusses some properties and problems of the used models and provides a conclusion and an outlook on future work. Appendix A presents selected results from preliminary research and Appendices B and C show detailed figures and scores obtained in the experiments.

2 Speech enhancement

The intention of this chapter is to provide the reader with the fundamental concepts of the fields of SE and SCSS. For this purpose we establish an abstract classification of different approaches and describe the most important ideas in detail in the following sections.

2.1 Fundamentals

Formally, SE is stated as the problem of computing an estimate $\hat{x}(t)$ of the clean speech out of a given signal

$$y(t) = x(t) + d(t), \tag{2.1}$$

where $y(t)$ is a additive mixture of the unknown clean speech signal $x(t)$ and some uncorrelated noise $d(t)$.

An arbitrary signal $s(t)$ can be transformed to the time-frequency domain by applying a discrete Fourier transform (DFT)

$$S^{\mathcal{C}}(k) \triangleq \mathrm{DFT}(s(t)) \qquad \text{(complex DFT coefficients)}, \tag{2.2}$$

where $k \in \{1, \ldots, K\}$ denotes the discrete frequency index. This can equivalently be expressed as

$$S^{\mathcal{C}}(k) = S(k) \cdot \frac{S^{\mathcal{C}}(k)}{S(k)} \tag{2.3}$$

$$= S(k) \cdot e^{i\phi^s(k)} \tag{2.4}$$

with

$$S(k) \triangleq |S^{\mathcal{C}}(k)| \qquad \text{(magnitude spectrum)}, \tag{2.5}$$

$$\phi_s(k) \triangleq \arg(S^{\mathcal{C}}(k)) \qquad \text{(phase)}. \tag{2.6}$$

When a signal is highly non-stationary a more appropriate representation is the discrete short time Fourier transform (STFT)

$$S^{\mathcal{C}}(n,k) \triangleq \mathrm{STFT}(s(t)) \qquad \text{(complex STFT coefficients)} \tag{2.7}$$

where $n \in \{1, \ldots, N\}$ denotes the frame index. This can be achieved by sequentially computing the DFT coefficients of short segments of the signal (windowing), during which the properties of the signal remain approximately constant. Equivalently to Equation 2.4, $S^{\mathcal{C}}(n,k)$ can be expressed as

$$S^{\mathcal{C}}(n,k) = S(n,k) \cdot e^{i\phi^s(nk)} \tag{2.8}$$

with

$$S(n,k) \triangleq |S^{\mathcal{C}}(n,k)| \qquad \text{(magnitude spectrogram}^1\text{)} \tag{2.9}$$

$$\phi_s(n,k) \triangleq \arg(S^{\mathcal{C}}(n,k)) \qquad \text{(phase)} \tag{2.10}$$

Thus, the fundamental SE equation 2.1 can be written in the time-frequency domain as

$$Y^{\mathcal{C}}(n,k) = X^{\mathcal{C}}(n,k) + D^{\mathcal{C}}(n,k) \tag{2.11}$$

and the goal of every SE algorithm is to compute an estimate $\hat{X}^{\mathcal{C}}(n,k)$ [1, pp. 13ff, 93f].

In general we will refer to Y as *mixture* or *noisy*, to X as *speech* or *clean* and to D as *noise*. X and D are denoted *sources* or *targets* whereas $\hat{X}$ and $\hat{D}$ are referred to as speech or noise *estimates*.

2.1.1 Separation of a signal mixture

Let X and Y denote an arbitrary representation of clean and noisy speech in the frequency domain. Every SE algorithm formulated in this domain can be described by the fundamental equation

$$\hat{X} = H \cdot Y \tag{2.12}$$

where H can be seen as a transformation or filter that is applied on the noisy mixture. How H is calculated, and what range it is defined over depends on the research field and the specific algorithm.

Algorithms that mainly originate from the field of signal processing (Section 2.2) usually define H as $H(k)$ and denote it as *gain-* or *suppression function*. A gain-function specifically denotes the *formula* by which H is calculated and is usually bound between 0 and 1. Machine-learning approaches (Section 2.4) usually define H as $H(n,k)$ and denote one concrete instantiation of H as *mask*. A *model* is any estimator that makes predictions based on the mixture. Given these definitions, we can identify three main approaches in the literature[2]

Approach	Model type	Description
DT	**Direct target** estimator	$\hat{X}$ is directly estimated from the mixture, H is not explicitly calculated[3]
DM	**Direct mask** estimator	H is directly estimated
IM	**Indirect mask** estimator	H is an explicitly defined gain-function where its components are estimated separately.

Table 2.1: The three different SE approaches, see Figure 2.1 for a graphical scheme.

[1] The expression *spectrogram* is not used consistently in the literature. While some refer to the magnitude $|S^{\mathcal{C}}(n,k)|$, others refer to the estimated power spectral density $|S^{\mathcal{C}}(n,k)|^2$ (often also called the periodogram). Furthermore it sometimes denotes the explicit graphical representation of the log-magnitude $\log(|S^{\mathcal{C}}(n,k)|)$.

[2] Although early SE studies do not use the term 'mask', they can be equally subsumed under the presented concepts.

Henceforth, we will classify all algorithms by their respective approach, which can be either DT, DM or IM as shown in Table 2.1 and Figure 2.1.

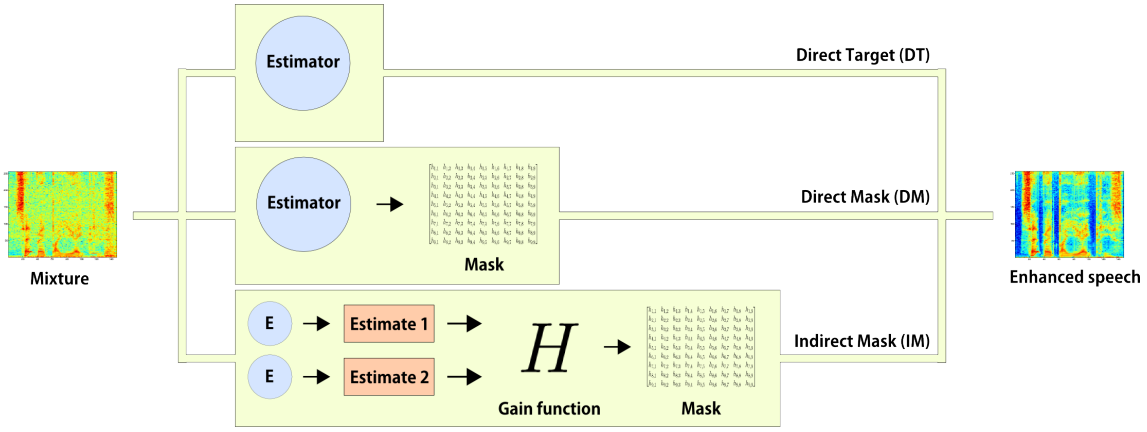


Figure 2.1: The three different SE approaches

The remainder of the chapter is structured as follows: Section 2.2 introduces important early SE approaches which mainly differ in the definition of their gain function and require a separate noise estimate (Section 2.3) to be functional (IM approach). Section 2.4 describes more recent SE approaches and the chapter is finished by shortly introducing evaluation techniques and discussing problems and challenges of the field (Sections 2.5 and 2.6).

[3] Of course it can be trivially calculated by $H = \hat{X}/Y$.

2.2 Amplitude estimators

The intention of this section is not to rigorously derive all presented equations but to give the reader an introduction of the results obtained in selected past studies and to present the main concepts and ideas that are crucial for understanding recent developments in SE. To that extent a short survey over selected early SE approaches originating from the domain of signal processing is presented. It is important to note that these are merely derivations of amplitude estimators (estimators of $\mathcal{X}$) and their respective gain functions ($\mathcal{H}$) that rely on separate estimates to be functional (IM approach). These separate estimates are discussed in Section 2.3.

All estimators presented in this section are derived by making statistical assumptions about the sources of the mixture but the estimators impose no restrictions on the required separate estimates. Early SE approaches however used them exclusively in a *non model-based* context.

2.2.1 Spectral subtraction algorithms

Spectral-subtractive algorithms (SSAs) were one of the first types of algorithms proposed for SE [11]. They are computationally simple as the main computations involved are a Fourier and an inverse Fourier transform and they are therefore well suited for real-time implementations. The basic idea behind SSAs is that an estimate of the noise spectrum $\hat{D}(k)$ can be subtracted from the noisy spectrum $Y(k)$ yielding an estimate of the clean speech spectrum $\hat{X}(k)$. Combined with the noisy phase this gives

$$\hat{X}^{\mathcal{C}}(k) = (Y(k) - \hat{D}(k)) \cdot e^{i\phi^y(k)} \tag{2.13}$$

which can be transformed to $\hat{x}(t)$ via an inverse Fourier transform. For rewriting Equation 2.13 as a gain function we note that a non-complex H can be seen a symmetric weighting of the real and imaginary parts of $Y^{\mathcal{C}}$. When we define H to act on the magnitude Y and use the noisy phase as an estimate of the true phase we see that this is equal to directly applying H on $Y^{\mathcal{C}}$

$$\begin{aligned} \hat{X} &= H \cdot Y, \\ \hat{X} \cdot \frac{Y^{\mathcal{C}}}{Y} &= H \cdot Y \cdot \frac{Y^{\mathcal{C}}}{Y}, \\ \hat{X}^{\mathcal{C}}_{\phi_y} &= H \cdot Y^{\mathcal{C}}. \end{aligned} \tag{2.14}$$

The gain function can then be given as

$$H(k) = 1 - \frac{\hat{D}(k)}{Y(k)}. \tag{2.15}$$

Per definition the magnitude spectrum cannot be negative. Since this might be the case when $\hat{D}(k) > Y(k)$, non-negativity can be ensured e.g. by using half-wave rectification, i.e. clamping all values below zero to zero. This however often leads to speech distortions in the form of musical noise (see 2.6) which have to be combated e.g. by oversubstraction [11].

The performance of SSAs was extensively evaluated [1, pp. 105ff,130ff], but mostly objective measures as the improvement in SNR or spectral similarity were used. These measures are moderately correlated with speech quality and intelligibility but formal listening tests suggest that this class of algorithms mainly improves speech quality. The impact on intelligibility was not significant and for very aggressive subtraction setups, intelligibility could even decrease.

2.2.2 Linear estimators / Wiener filters

The Wiener filtering approach is a widely used class of filters in SE history [12] [1, 137ff]. They are optimal in the sense that they minimize a mathematically tractable error criterion. In this setup the noisy mixture $y(t)$ passes through a linear, time-invariant filter that produces the output signal $\hat{x}(t)$ in such a way that the estimated error $e(t) = x(t) - \hat{x}(t)$ is minimal. The output signal is formed as a convolution of the filters impulse response $h(t)$ and the input signal $y(t)$. Formulated in the frequency domain this gives

$$\hat{X}^{\mathcal{C}}(k) = H^{\mathcal{C}}(k)Y^{\mathcal{C}}(k) \tag{2.16}$$

with the error function

$$E^{\mathcal{C}}(k) = X^{\mathcal{C}}(k) - \hat{X}^{\mathcal{C}}(k). \tag{2.17}$$

Under the assumption that noise and speech are uncorrelated, minimizing the mean-square error $\mathbb{E}[E^2(k)]$ yields the optimal filter

$$H^{\mathcal{C}}(k) = \frac{\lambda^x(k)}{\lambda^x(k) + \lambda^d(k)}, \tag{2.18}$$

where

$$\lambda^x(k) \triangleq \mathbb{E}\left[X^2(k)\right] \qquad \text{(speech power spectrum)}, \tag{2.19}$$

$$\lambda^d(k) \triangleq \mathbb{E}\left[D^2(k)\right] \qquad \text{(noise power spectrum)} \tag{2.20}$$

represent the power spectra of the mixture $y(t)$ and noise $d(t)$ respectively. Power spectra are real and nonnegative, thus $H^{\mathcal{C}}(k)$ is also real and nonnegativ and we can write it as $H(k)$. Equation 2.18 can be rewritten as

$$H(k) = \frac{\xi(k)}{\xi(k) + 1}, \tag{2.21}$$

where

$$\xi(k) \triangleq \frac{\lambda^x(k)}{\lambda^d(k)} \qquad (\textit{a priori} \text{ SNR}). \tag{2.22}$$

From Equation 2.21 follows that in regions of low SNR, where $\xi(k) \longrightarrow 0$, the attenuation is high ($H^{\mathcal{C}}(k) \approx 0$) and for high SNRs $\xi(k) \longrightarrow \infty$ the attenuation is low ($H^{\mathcal{C}}(k) \approx 1$).

In order to estimate the unknown component $\xi(k)$ two mayor ways have been proposed: (1) By estimating the clean speech power spectrum (e.g. by iterative Wiener filtering, compare [1, pp. 156ff] for a detailed description) (2) By directly estimating the *a priori* SNR $\xi(k)$, for instance with the "decision-directed" approach described in Section 2.3.3.

It has to be noted that the Wiener filters are derived under the assumption that the input signal is stationary, which is certainly not true for speech and often not true for noise signals. Furthermore, they are only optimal in the linear sense and thus limited to applying linear transformation to the input signal. Non-linear filters are less constraint and can potentially yield better performance.

2.2.3 Non-linear estimators

Non-linear estimators are usually defined over the magnitude $Y(k)$ and not over the complex spectrum $Y^{\mathcal{C}}(k)$. They make explicit assumptions about the speech and noise magnitude distri-

butions and often employ additional gain modifications to take speech presence or absence into account. The SE problem in this variant assumes that the probability distribution of the noisy speech magnitude $Y(k)$ is parametrized by the unknown set of parameters θ, that represent the clean speech magnitude. The task is then to estimate a configuration or probability distribution of θ that fits well to the observed data. For resynthesis, commonly the noisy phase $\phi^y(k)$ is used, as it can be derived that in the sense of a minimum mean square error (MMSE) estimator, the noisy phase is an optimal estimate of the clean phase [13].

Maximum-likelihood (ML) estimator

This approach follows the frequentist interpretation of probabilities, i.e. θ is an unknown but deterministic parameter. The maximum likelihood (ML) formulation thus can be written as

$$\hat{\theta}_{\mathrm{ML}} = \arg\max_{\theta} \mathrm{P}(Y^{\mathcal{C}}(k);\, \theta). \tag{2.23}$$

Under the assumption that $D^{\mathcal{C}}(k)$ is a zero-mean, complex Gaussian and its real and imaginary parts have the variances $\lambda^d(k)/2$ the probability density function of the mixture is also a Gaussian

$$P\big(Y^{\mathcal{C}}(k);\, X(k),\, \phi^x(k)\big) = \frac{1}{\pi\lambda^d(k)} \exp\left[-\frac{|Y^{\mathcal{C}}(k) - X(k)e^{i\phi^x(k)}|^2}{\lambda^d(k)}\right]. \tag{2.24}$$

By integrating out the phase $\phi^x(k)$ and differentiating the log-likelihood function of the above distribution with respect to $X(k)$, the maximum likelihood solution of the clean speech magnitude spectrum is given by

$$\hat{X}(k) = \frac{1}{2}\left[Y(k) + \sqrt{Y^2(k) - \lambda^d(k)}\right], \tag{2.25}$$

thus the clean speech estimate only depends on the noisy power spectrum and an estimate of the noise. Formulated as a gain function this gives

$$H(k) = \frac{1}{2} + \frac{1}{2}\sqrt{\frac{Y^2(k) - \lambda^d(k)}{Y^2(k)}}. \tag{2.26}$$

As this approach does not provide sufficient attenuation by itself, in [12] a two-state decision model of speech presence and absence was used.

Minimum mean squared error (MMSE) estimator

The MMSE estimator adopts a Bayesian perspective on estimating the speech magnitude by incorporating prior knowledge about θ by defining a prior distribution over $X(k)$. The MSE between the estimated and the true magnitude is given by

$$\mathbb{E}[(\hat{X}(k) - X(k))^2], \tag{2.27}$$

where the expectation is towards the joint probability $\mathrm{P}(X(k), Y(0), ..., Y(K))$. Under the assumption of statistical independence of the DFT coefficients of $Y(k)$, minimizing 2.27 with respect to $X(k)$ gives

$$\hat{X}(k) = \mathbb{E}[X(k) \,|\, Y^{\mathcal{C}}(k)], \tag{2.28}$$

which is the expectation of the posterior. By applying Bayes theorem

$$\mathrm{P}(X(k)\,|\,Y^{\mathcal{C}}(k)) = \frac{\mathrm{P}(Y^{\mathcal{C}}(k)\,|\,X(k))\,\mathrm{P}(X(k))}{\mathrm{P}(Y^{\mathcal{C}}(k))} = \frac{\mathrm{P}(Y^{\mathcal{C}}(k)\,|\,X(k))\,\mathrm{P}(X(k))}{\int \mathrm{P}(Y^{\mathcal{C}}(k)\,|\,x(k))\,\mathrm{P}(x(k))\,dx(k)} \tag{2.29}$$

it can be seen that the estimator is a function of the likelihood $\mathrm{P}(Y^{\mathcal{C}}(k)\,|\,X(k))$ and the prior distribution $\mathrm{P}(X(k))$. The assumed statistical model requires that $Y^{\mathcal{C}}(k)$ is a sum of two complex Gaussians and by integrating over the phase we arrive at the same model as given in Equation 2.24. $X(k)$ is Rayleigh distributed since it is defined as $X(k) = |X^{\mathcal{C}}(k)| = \sqrt{\Re(X^{\mathcal{C}}(k))^2 + \Im(X^{\mathcal{C}}(k))^2}$, where $\Re$ and $\Im$ denote the real and imaginary parts of a complex number. With these distributions the optimal MMSE magnitude estimator can be derived and is given by

$$\hat{X}(k) = \left[\frac{\sqrt{v(k)}}{\gamma(k)}\Gamma(1.5)\Theta(-0.5, 1;\ -v(k))\right] Y(k), \tag{2.30}$$

where $\Gamma(\cdot)$ denotes the gamma function, $\Theta(\cdot)$ denotes the confluent hypergeometric function, and

$$v(k) \triangleq \gamma(k)\frac{\xi(k)}{1+\xi(k)} \tag{2.31}$$

is a function of the *a priori* SNR $\xi(k)$ defined in Equation 2.22 and

$$\gamma(k) \triangleq \frac{Y^2(k)}{\lambda^d(k)} \qquad (\textit{a posteriori}\ \text{SNR}). \tag{2.32}$$

The MMSE approach thus depends on the *a priori* SNR, that can be seen as the unknown, true SNR, and the *a posteriori* SNR, which is the observed SNR, given a noise estimate. The gain function follows immediately from 2.30.

2.3 Noise and SNR estimators

Amplitude estimators or their respective gain functions require a separate noise estimate or an estimate of the *a priori* SNR ξ. Therefore, the estimators presented in this section are not SE approaches on their own but need to be combined with a specific amplitude estimator (Section 2.2) or form part of a more advanced setup (see Section 2.4).

2.3.1 Noise estimation

Early noise estimation techniques were limited to estimating the noise during periods of non-speech activity, employing a voice activity detection (VAD) algorithm. A VAD algorithm tries to determine the presence or absence of speech, based on features extracted from the noisy signal (e.g. ceptral features, Itakura LPC spectral distance). The noise is then re-estimated during speech pauses which also occur during continuous speech, mainly as a short silence at the start of a plosive (e.g. /p/, /t/, /k/).

Evidently, these approaches still suffer from the inability to detect changes in the noise level during the articulation of non-plosive phonemes and also encounter problems in low SNR conditions [1, pp. 377]. Therefore noise-estimation algorithms with the ability to cope with non-stationary noise, i.e. with the ability to estimate the noise even during arbitrary speech periods, were developed.

According to [1, pp. 380], most of these algorithms can be divided into three main classes: (1) histogram-based, (2) minimum-tracking and (3) time-recursive averaging. The histogram-based algorithms stem from the observation, that the mode of the histogram of the energy values in one frequency band often corresponds to the true noise level. This pattern however is not consistent over all frequency bands. Minimum-tracking and time-recursive averaging algorithms are the basic blocks for the IMCRA algorithm used in the practical part of this work and are described in the next sections.

Minimum statistics

Minimum statistics (MS) are based on the observation that the power of the noisy signal in each frequency band tends to decay to the true noise power even during speech activity [14]. This is due to the fact that speech is in general sparse [3], thus a noise estimate can be found be tracking the minimum power values of the noisy signal within a window of defined length under the assumption that the noise and the speech are statistically independent, i.e. $Y^2(n,k) \approx X^2(n,k) + D^2(n,k)$. The minimum for a window of length W is then defined as

$$P_{min}(n,k) = \min\left\{Y^2(n,k), Y^2(n-1,k), \ldots, Y^2(n-W+1,k)\right\}. \tag{2.33}$$[4]

As for nontrivial probability distributions the minimum is smaller than the mean, P_{min} consequently underestimates the true noise level. To compensate for this, a bias compensation factor B_{min} can be derived, so that the noise power estimate is then given by

$$\hat{\lambda}^d(n,k) = B_{min}(n,k) \cdot P_{min}(n,k). \tag{2.34}$$

Recursive averaging

Time- and frequency dependent recursive averaging also makes use of the fact that speech is sparse and depending on the current phoneme, individual frequency bands have different effective SNRs which are used to update each frequency band individually. E.g. while producing fricatives where the spectral power is mainly located in the high frequency bands, the noise estimate can be reliably updated for the low-frequency bands. The averaging happens time-recursively and directly provides an estimate for the noise power spectrum

$$\hat{\lambda}^d(n,k) = \alpha(n,k)\hat{\lambda}^d(n-1,k) + (1-\alpha(n,k))Y^2(n,k). \tag{2.35}$$

Thus, whenever $\alpha(n,k) \approx 1$ the noise spectrum is not updated and the previous value is used. When $\alpha(n,k) \approx 0$ the current spectral power of $Y^2(n,k)$ becomes the new noise estimate.

Different methods of estimating the smoothing constant $\alpha(n,k)$ exist. One method estimates the parameter as a sigmoid function of an estimate of the *a posterior* SNR [15], another method only updates the estimate when the estimate of the *a posterior* SNR passes a predefined threshold [16].

A statistical approach to estimating the noise power spectrum is given by minimizing the MSE $\mathbb{E}[(\hat{\lambda}^d(n,k) - D^2(n,k))^2]$ under the two hypothesis

$$\mathcal{H}^0 : Y^{\mathcal{C}}(n,k) = D^{\mathcal{C}}(n,k) \qquad \text{(speech absent)}, \tag{2.36}$$

$$\mathcal{H}^1 : Y^{\mathcal{C}}(n,k) = X^{\mathcal{C}}(n,k) + D^{\mathcal{C}}(n,k) \qquad \text{(speech present)}, \tag{2.37}$$

[4] In [14], an already smoothed version of Y^2 is used to calculate P_{min}

which yields

$$\begin{aligned}\hat{\lambda}^d(n,k) = &\ \mathbb{E}\left[\lambda^d(n,k)\,|\,Y^{\mathcal{C}}(n,k)\right] \\ = &\ \mathbb{E}\left[\lambda^d(n,k)\,|\,\mathcal{H}^0\right] P(\mathcal{H}^0\,|\,Y^{\mathcal{C}}(n,k)) \\ &+ \mathbb{E}\left[\lambda^d(n,k)\,|\,\mathcal{H}^1\right] P(\mathcal{H}^1\,|\,Y^{\mathcal{C}}(n,k)).\end{aligned} \tag{2.38}$$

When the conditional probabilities of the above equation are identified with the smoothing factor $\alpha(n,k)$ and the expectations are approximated by $\hat{\lambda}^d(n-1,k)$ and $Y^2(n,k)$ respectively, Equation 2.38 takes the same form as Equation 2.35. Thus, recursive averaging also emerges in the context of the hypothesis about speech presence and absence.

2.3.2 IMCRA noise estimator

The improved minima controlled recursive averaging (IMCRA) algorithm combines the robustness of minimum tracking with the simplicity of recursive averaging [17]. We take as a starting point two Gaussian models of the noisy power spectrum that depend on the hypothesis about speech presence and absence as defined in Equations 2.36 and 2.37:

$$\mathrm{P}\big(Y(n,k)\,|\,\mathcal{H}^0(n,k)\big) = \frac{1}{\pi\lambda^d(n,k)} \exp\left[-\frac{Y^2(n,k)}{\lambda^d(n,k)}\right], \tag{2.39}$$

$$\mathrm{P}\big(Y(n,k)\,|\,\mathcal{H}^1(n,k)\big) = \frac{1}{\pi(\lambda^x(n,k)+\lambda^d(n,k))} \exp\left[-\frac{Y^2(n,k)}{\lambda^x(n,k)+\lambda^d(n,k)}\right]. \tag{2.40}$$

The conditional speech presence probability

$$p(n,k) \triangleq \mathrm{P}\big(\mathcal{H}^1(n,k)\,|\,\gamma(n,k)\big) \qquad \text{(speech presence probability)} \tag{2.41}$$

is derived as

$$p(n,k) = \left[1 + \frac{q(n,k)}{1-q(n,k)}(1+\xi(n,k)) \exp\left[-v(n,k)\right]\right]^{-1}, \tag{2.42}$$

where in analogy to Equation 2.31

$$v(n,k) \triangleq \gamma(n,k)\frac{\xi(n,k)}{1+\xi(n,k)} \tag{2.43}$$

and

$$q(n,k) \triangleq \mathrm{P}(\mathcal{H}^0(n,k)) \qquad (\textit{a priori} \text{ speech absence probability}). \tag{2.44}$$

Under the assumption that the noise estimate should be updated via recursive averaging when speech is absent and held when speech is present, the noise estimate is given by

$$\hat{\lambda}^d(n,k) = (1-p(n,k))\left[\alpha\lambda^d(n-1,k) + (1-\alpha)Y^2(n,k)\right] + p(n,k)\left[\lambda^d(n-1,k)\right], \tag{2.45}$$

which can be simplified to

$$\hat{\lambda}^d(n,k) = \alpha^d(n,k)\left[\hat{\lambda}^d(n-1,k)\right] + (1-\alpha^d(n,k))\left[Y^2(n,k)\right], \tag{2.46}$$

where

$$\alpha^d(n,k) \triangleq \alpha + (1-\alpha)p(n,k). \tag{2.47}$$

It can be seen that the noise estimation only depends on the speech absence probability $q(n,k)$ and the *a priori* SNR $\xi(n,k)$.

For estimating $q(n,k)$ an estimator depending on the minima values of the smoothed power spectrum of the noisy speech is developed. The smoothing is carried out in time and frequency and happens in two iteration where the first iteration provides a rough voice activity detection and the second excludes strong speech components to make the minimum tracking robust [17]. Given a smoothing factor $\tilde{\alpha}$ and a normalized window function $w(\cdot)$ of the length $2L+1$, the first frequency smoothing is given by

$$S^f(n,k) = \sum_{i=-L}^{L} w(i) \cdot Y^2(n,k-i) \tag{2.48}$$

and the time smoothing is performed by successive recursive averaging

$$S(n,k) = \tilde{\alpha} S(n-1,k) + (1-\tilde{\alpha}) S^f(n,k). \tag{2.49}$$

The minimum value $S^{min}(n,k)$ of $S(n,k)$ is calculated equivalently to Equation (2.33) and a bias compensation factor B^{min} is estimated [17]. Based on the two ratios

$$\gamma^{min}(n,k) \triangleq \frac{Y^2(n,k)}{B^{min} S^{min}(n,k)} \qquad \text{(MS-based } a \textit{ posteriori} \text{ SNR)}, \tag{2.50}$$

$$\zeta(n,k) \triangleq \frac{S(n,k)}{B^{min} S^{min}(n,k)} \qquad \text{(smoothed MS-based } a \textit{ posteriori} \text{ SNR)} \tag{2.51}$$

the rough VAD of the first smoothing iteration is given by

$$I(n,k) = \begin{cases} 1, & \text{if } \gamma^{min}(n,k) < \gamma^0 \text{ and } \zeta(n,k) < \zeta^0 \\ 0, & \text{otherwise} \end{cases}, \tag{2.52}$$

where γ^0 and ζ^0 are two thresholds. This can be interpreted as that speech is detected ($I(n,k) = 0$) whenever neither the current nor the smoothed noisy spectrum energies are high in comparison to the current MS-estimates.

For the second smoothing iteration we define

$$\tilde{S}^{\tilde{f}}(n,k) = \frac{\sum_{i=-L}^{L} w(i) \cdot I(n,k-i) \cdot Y^2(n,k-i)}{\sum_{i=-L}^{L} w(i) \cdot I(n,k-i)} \tag{2.53}$$

within a given window $w(\cdot)$, which is a normalized sum of all values of the noisy power spectrum where speech is absent according to $I(n,k)$. The second frequency smoothing is then given by

$$\tilde{S}^f(n,k) = \begin{cases} \tilde{S}^{\tilde{f}}(n,k), & \text{if } \sum_{i=-L}^{L} I(n,k-i) \neq 0 \\ \tilde{S}^f(n-1,k), & \text{otherwise} \end{cases}, \tag{2.54}$$

which corresponds to using $\tilde{S}^{\tilde{f}}(n,k)$ when speech is at least one time absent within the given window and the value of the last smoothed frame otherwise. The smoothing in time is given

equivalently to Equation 2.49 by

$$\tilde{S}(n,k) = \tilde{\alpha}\tilde{S}(n-1,k) + (1-\tilde{\alpha})\tilde{S}^f(n,k). \tag{2.55}$$

By excluding the high-energy speech components minimum tracking gets more reliable and the search window length can be reduced which yields faster responses to changes in the noise floor. The estimator for the *a priori* speech absence probability is then given by

$$\hat{q}(n,k) = \begin{cases} 1, & \text{if } \tilde{\gamma}^{min}(n,k) \leq 1 \text{ and } \tilde{\zeta}(n,k) < \zeta^0 \\ \frac{\gamma^1 - \tilde{\gamma}^{min}(nk)}{\gamma^1 - 1}, & \text{if } 1 < \tilde{\gamma}^{min}(n,k) < \gamma^1 \text{ and } \tilde{\zeta}(n,k) < \zeta^0 \\ 0 & \text{otherwise} \end{cases}, \tag{2.56}$$

where $\tilde{\gamma}^{min}(n,k)$ and $\tilde{\zeta}(n,k)$ are defined as in Equations 2.50 and 2.51 with $\tilde{S}(n,k)$ and $\tilde{S}^{min}(n,k)$ instead of $S(n,k)$ and $S^{min}(n,k)$. This estimator thus assumes that speech is absent ($\hat{q}(n,k) = 1$) whenever both SNR-measures are below a certain threshold and predicts speech presence when one or both are above. In between, the estimator provides a soft transition corresponding to its uncertainty about speech presence or absence.

According to Equation 2.42 we also need to estimate the *a priori* SNR where in [17] a modified version of the "decision-directed" approach (Equation 2.61) is used.

2.3.3 A priori SNR estimation

For estimating the *a priori* SNR, [18] developed a maximum likelihood (ML) approach and a more important "decision-directed" estimator, which will be described next: The *a priori* SNR can be expressed in terms of the *a posteriori* SNR as following:

$$\begin{aligned} \xi(n,k) &= \frac{\mathbb{E}\left[X^2(n,k)\right]}{\lambda^d(n,k)} = \frac{\mathbb{E}\left[Y^2(n,k) - D^2(n,k)\right]}{\lambda^d(n,k)} = \frac{\mathbb{E}\left[Y^2(n,k)\right]}{\lambda^d(n,k)} - \frac{\mathbb{E}\left[D^2(n,k)\right]}{\lambda^d(n,k)} \\ &= \mathbb{E}\left[\gamma(n,k)\right] - 1, \end{aligned} \tag{2.57}$$

where in analogy to Equations 2.22 and 2.32 the SNRs are given by

$$\xi(n,k) \triangleq \frac{\lambda^x(n,k)}{\lambda^d(n,k)} \triangleq \frac{\mathbb{E}[X^2(n,k)]}{\mathbb{E}[D^2(n,k)]} \qquad (a \textit{ priori} \text{ SNR}), \tag{2.58}$$

$$\gamma(n,k) \triangleq \frac{Y^2(n,k)}{\lambda^d(n,k)} \qquad (a \textit{ posteriori} \text{ SNR}). \tag{2.59}$$

Now we can write the *a prior* SNR as a combination of Equations 2.58 and 2.57, i.e.

$$\xi(n,k) = \mathbb{E}\left[\frac{1}{2}\frac{X^2(n,k)}{\lambda^d(n,k)} + \frac{1}{2}(\gamma(n,k) - 1)\right]. \tag{2.60}$$

This calculation can be made recursive by replacing $1/2$ with a weighting factor α_{DD} and the first term of the sum with the estimates obtained in the previous frame

$$\hat{\xi}(n,k) = \alpha_{DD}\frac{\hat{X}^2(n-1,k)}{\lambda^d(n-1,k)} + (1-\alpha_{DD})\max(\gamma(n,k) - 1, 0). \tag{2.61}$$

The $\max(\cdot)$ operator is used to ensure non-negativity.

This estimator was widely adopted in the literature and provided the basis for various improvements (see [1, pp. 219]) but most notably is capable of suppressing musical noise.

2.4 Model-based approaches

Recent advances in SCSS and SE were achieved by applying state-of-the-art machine learning techniques that make use of source-adapted statistical models. On contrary to methods discussed in Sections 2.2 and 2.3, machine-learning techniques explicitly model the distribution of the sources in a preceding training step. This can be seen as statistical pattern recognition [3] which uses acoustic grouping cues to build adapted models of the input data.

Following the common distinctions in the machine learning field, every probabilistic learning algorithm can be classified by 2 criteria: (1) Modelled distribution and (2) training supervision. Let X and Y be two random variables where both are observed for training and only Y is observed for testing. Under this assumption, regarding (1), a model is called *discriminative* when it models the conditional distribution $\mathrm{P}(X|Y)$ and *generative* if it models $\mathrm{P}(Y)$, $\mathrm{P}(X)$ or $\mathrm{P}(X,Y)$. Regarding (2), a training algorithm is called *supervised* if X is given as a label of Y and *unsupervised* if only Y is given[5]. In SE this corresponds to the distinction between models, that (usually discriminatively and supervised) learn some kind of mapping, or generatively model a distribution over the sources, combining them by means of an interaction model.

Binary masks and softmasks

Historically binary masks first appeared in the field of CASA (computational auditory scene analysis) and were introduced to SCSS as a mere simplification in [3,7]. Nevertheless, according to [2] and [1, pp. 613ff], one contributing factor to the success of more recent SCSS approaches was precisely the adoption of binary masks. Although the algorithms in the former sections can yield significant increase in perceived speech quality, they are still incapable of increasing speech intelligibility [1, pp. 609ff]. A study [19] conducted to test if an ideal Wiener gain function (with access to the true SNRs) can improve speech intelligibility concluded that this is possible, but that the distribution of the values of the ideal gain function is bimodal, with one mode at 0dB and the other <-27dB. That is, or heavy attenuation is applied or the frequency bin is left untouched. This suggests that the ideal Wiener gain function is effectively not a soft-gain but rather a binary function, which motivates the use of binary masks retaining only bins with favourable SNR and discarding the other ones. Formally this can be written as

$$H(n,k) = \mathrm{BM}(n,k) \triangleq \begin{cases} 1 & \text{if } C(n,k) > \gamma \\ 0 & \text{otherwise} \end{cases} \qquad \text{(binary mask)}, \tag{2.62}$$

where $C(n,k)$ is an algorithm-dependent channel-selection criterion and γ is a threshold.

For the ideal binary mask (IBM) the most common channel selection criterion is the true SNR ($C(n,k) = \xi(n,k)$) with $\gamma = 0$. It was shown [1, 3] that using binary masks instead of real-valued soft-masks does not significantly lower the upper bound of achievable performance in intelligibility but instead facilitates the estimation process. Because of good empirical results and its versatility, Wang [2] proposed the IBM as the ultimate computational goal of CASA, which could provide SE and SCSS approaches with a common ground-truth.

Despite the fact that much recent research appears to be focused on estimating BMs, it is still unclear whether BMs should always be preferred over a continuous valued representation of $H(n,k)$. Peharz et al. [20] define a continuous mask (CM) in the power spectral domain which is equivalent to the Wiener gain function given in 2.18. Results with this mask suggest that it depends on the objective whether BMs or CMs should be used. Zöhrer et al. [6, 10] use a

[5] From this follow a couple of combinations, but the most common models are either discriminative and supervised (used for classification and regression) or generative and unsupervised.

continuously valued softmask (SM) which is defined as

$$H(n,k) = \mathrm{SM}(n,k) \triangleq \frac{\hat{X}(n,k)}{\hat{X}(n,k) + \hat{D}(n,k)} \qquad \text{(estimated softmask)}, \tag{2.63}$$

which is the same as the CM but in the magnitude domain. Although [10] give scores for binary masks, relevant quality measures were only given for the SM-resynthesized utterances, indicating that the softmask approach might has been more successful in this setup. The ideal softmask in this setup is defined as

$$H(n,k) = \mathrm{ISM}(n,k) \triangleq \frac{X(n,k)}{X(n,k) + D(n,k)} \qquad \text{(ideal softmask)}. \tag{2.64}$$

It has to be noted that the IM-approach of [6] only makes sense in case of two *separate* estimates of speech and noise. Otherwise the result is equal to directly estimating the speech as

$$\hat{D} = Y - \hat{X}, \tag{2.65}$$

$$H = \frac{\hat{X}}{\hat{X} + Y - \hat{X}} = \frac{\hat{X}}{Y}, \tag{2.66}$$

$$\hat{X} = \frac{\hat{X}}{Y} \cdot Y = \hat{X}. \tag{2.67}$$

2.4.1 Linear models

Early approaches of incorporating source-adapted models into the SE domain were built on well-established statistical models such as vector quantizers (VQs) or Gaussian mixture models (GMMs). They follow an unsupervised and generative approach, i.e. all sources are modelled separately and inference is done by inferring the most likely states of the source-models given an input sequence and a defined mixture model.

The log-max approach

Roweis [3] was the first one to make use of the well known fact that unless both numbers e_1 and e_2 are very close, the logarithm of the sum of two numbers is approximately the maximum of the logarithm of each number, that is

$$\log(e_1 + e_2) \approx \max\left(\log e_1, \log e_2\right). \tag{2.68}$$

Applied to the problem of SE this means, that unless the sources are highly dependent, the log-spectrogram of the mixture is approximately the maximum of the individual log-spectrograms. This observation gave rise to the MAXVQ [7] and the initial FHMM [3] model, where both use the log-max approximation to keep the computation feasible.

The MAXVQ [7] model is a very simple model as it does not incorporate temporal dependencies but trains frame-wise vector quantizers (VQ) for each source which are then combined to explain the mixture. Trivial inference can be achieved by comparing every possible combination of VQ-outputs with the current observation but with the log-max approximation an upper bound for the probability of the proposed output given the the state of one VQ can be derived and a branch-and-bound technique for efficient MAP-inference is inferred.

Adaptive Wiener filtering

The intention of adaptive Wiener Filtering [4,21,22] is to extend Wiener Filtering to the domain of GMMs, where one GMM is trained for each source. If we condition on one component of the GMM, we can perform Wiener Filtering exactly as defined in Equation 2.18. Adaptive Wiener Filtering can now be seen as performing Wiener Filtering for every combination of Gaussians in the source-models, which is weighted by the probability of an algorithm-specific interaction model:

$$\hat{X}^{\mathcal{C}}(n,k) = \sum_{i,j} \gamma_{ij}(n) \frac{\lambda^{x_i}}{\lambda^{x_i} + \lambda^{d_j}} Y^{\mathcal{C}}(n,k), \tag{2.69}$$

where i and j are the indices of the Gaussians and γ_{ij} is the weighting factor of the interaction model.

From this basic setting many extensions have been proposed, although they mainly originate from the SCSS domain. Tsai et al. [22] proposed a method of learning a voice and a music model directly from the music mixture. Ozerov et al. [4] take this approach even further by training a general voice model which is then adapted to the individual recording by assuming that the adapted model is obtained from the general voice model by a linear transformation. Benaroya et al. [21] train the source-models with a Gaussian scaled mixture model (GSMM) to account more efficiently for equal sounds with different amplitudes.

FHMM models

To account for the sequential nature of speech, often HMMs are incorporated. One can extend a single speaker HMM, used e.g. for phoneme recognition, to a factorial HMM (FHMM) [3,5,23] by implementing two separately evolving Markov chains, where their joint probability factorizes according to a given interaction model. For SE, one of the HMMs is trained on clean speech where the other is trained on noise.

The FHMM model served as starting point for many advanced developments in the field. Roweis [3] predicts binary separation masks using the log-max approach to upper-bound the probability of the observation given the state of one chain in order to efficiently calculate the best joint trajectory of both sources given the mixture. Hershey et al. [5] extended each HMM by an additional chain to model grammar dynamics, which ultimately led to a system that was able to outperform humans on a speech separation and recognition task with limited vocabulary and grammar [24]. Exact inference nevertheless still scales exponentially with the number of sources, thus various approximate inference methods especially suitable for SCSS settings with more than two speakers have been developed (see [25] for a detailed survey).

Due to the generality of the FHMM approach it is also possible to use it for other problems than source separation – for example Wohlmayr et al. [23] used an FHMM to track the pitch of two simultaneously active speakers including the assignment of individual pitch trajectories to each speaker.

2.4.2 Non-linear models

Non-linear models apply a non-linear function to its input. A common representative is the artificial neural network (ANN), which can have either a single layer or multiple layers. Multi-layer networks, also called deep models, are potentially very powerful but until recently [26], it was not possible to train them effectively.

Deep (non-linear) models have certain adantages over linear models: As speech is produced by modulating only very few physical parameters, it must lie on a low-dimensional manifold within

the high-dimensional representation of speech provided e.g. by a spectrogram. Unfortunately, linear models as e.g. a GMM may require a lot of components to adequately model even trivial, non-linear structures (e.g. a sphere) and furthermore often require the modelled coefficients to be roughly independent. Therefore often compressed speech representations as MFCCs are used, but this comes at the price of discarding information that might be important for specific problems as e.g. discrimination (see [27]). Deep Models on the other hand have the potential to learn much better models of data that lies close to a low-dimensional manifold and do not require the data to be statistically independent.

In the field of SCSS/SE, deep models have already been applied in various flavours. Xu et al. [28] took a straight-forward DT-approach and trained a deep neural network (DNN) to learn the direct mapping from noisy to clean speech. This was done by building a generative model consisting of stacked up restricted Boltzmann machines (RBMs) with subsequent discriminative fine-tuning. Wang and Wang [29] use a conditional random field (CRF) to model the temporal dynamics of speech, but employ a non-linear expansion in form of a pre-trained DNN, which is used as a non-linear preprocessor for the speech data. The CRFs are used as subband sequence classifier that provide an DM-estimate of the IBM.

Zöhrer and Pernkopf [10] evaluated the performance of generative stochastic networks (GSNs), deep believe networks (DBNs) and multi-layer perceptrons (MLPs) by directly predicting the softmask (DM-approach). In a follow-up study [6] four representation models (RBMs, conditional RBMs (CRBMs), higher order contractive autoenconders (HCAEs) and GSNs) were evaluated on the same task using an IM-approach.

SE can also be an important pre-processing step for automatic speech recognition systems (ASRs). Rennie et. al. [30] developed a noise robust speech recognizer by applying "factorial" inference on 2 RBMs yielding an factorial hidden restricted Bolzmann machine (FHRBM). Seltzer et al. [31] showed to increase robustness towards variabilities in the input by training a DNN with dropout [32].

2.5 Evaluation

SE algorithms can be evaluated towards two main evaluation criteria: perceptual quality and intelligibility. Perceptual quality is a subjective measure for evaluating the naturalness or "pleasantness" of the enhanced speech, whereas intelligibility measures the amount of words that can be identified correctly. To evaluate these two criteria, there exist two main ways: Subjective listening tests and evaluation by objective measures.

Subjective listening tests are conducted by having test subjects listen to enhanced speech samples under standardized conditions. The test subjects then rate the samples according to different criteria, e.g. intelligibility or quality. Many different methodologies exists but listening tests are a very-time consuming which is the reason why many SE studies merely rely on objective measures.

Objective measures try to quantify perceptual quality or intelligibility by measuring some numerical similarity between the original and the estimated signal. It is clear that for an objective measure to be suitable, it has to show high correlation with subjective listening tests.

2.5.1 Quality measures

Two widely used quality measures are the SNRs calculated in the time- or frequency domain:

$$\text{tSNR} = \frac{10}{T}\sum_{m=0}^{T-1}\log_{10}\frac{\sum_{t=Wm}^{Wm+W-1}x^2(t)}{\sum_{t=Wm}^{Wm+W-1}(x(t)-\hat{x}(t))^2}, \tag{2.70}$$

$$\text{fwSNR} = \frac{10}{N}\sum_{n=0}^{N-1}\frac{\sum_{k=1}^{K}w_k\log_{10}\left[X^2(n,k)/(X(n,k)-\hat{X}(n,k))^2\right]}{\sum_{k=1}^{K}w_k}, \tag{2.71}$$

where W is the frame length (typically 15-20ms), T is the number of frames in $x(t)$, N and K denote the total number of frequency bands and frames in $X(n,k)$ and w_k represents the weights placed on each frequency band.

A more advanced but very speech-specific approach are quality measures based on linear prediction coding (LPC) coefficients which efficiently encode speech by estimating the speech formants. These estimated coefficients are a representation of the speech frequency envelope and are stored together with the resulting residual signal which is calculated by subtracting the formant shape from the frequency envelope. The main idea of LPC-based measures as e.g. the Itakura-Saito distance is then to compare the coefficients of the clean speech with the coefficients of the estimated enhanced speech.

A study conducted in [1, pp. 513] showed that the tSNR only weakly correlates with the quality scores obtained in a subjective listening test. Although the fwSNR and Itakura-Saito distance measures showed higher correlations, neither of the measures yielded high correlation for a wide range of possible speech distortions [1, pp. 514f]. The highest correlation in this study was obtained by the PESQ-measure.

PESQ

PESQ [33] stands for perceptual evaluation of speech quality and is a widely used measure, even in current literature. It was developed to correctly assess the perceptional quality of enhanced speech, explicitly taking account of distortions that occur when speech is transmitted over telecommunication networks (e.g. codec distortions or signal delays).

First the reference and the enhanced signal are normalized to the same power level to make them comparable and run through a filter whose frequency response is approximately that of a standard telephone handset. In the next step, the signals are dynamically aligned to compensate for delays that might have been introduced by the transmission channel. This is done in a first step by calculating the cross-correlation of the two signals to crudely estimate the delay of different samples. In a second step the signals are dived into subsections which are realigned to optimize the matching, yielding an optimal division and alignment. The aligned signals are then processed by an auditory transform which mimics the key properties of human hearing, effectively removing parts that are inaudible to a listener. From the transformed signals a disturbance signal is calculated where the weighting of the error depends on its type. That is, positive and negative loudness differences are treated differently. The disturbance signals is finally converted to a PESQ score, which ranges from 1 (bad) to 4.5 (excellent).

Although PESQ yielded the highest correlations in the study of [1, pp. 514f], [34] reportes only moderate accuracy values and developed an even better measure called PEASS.

PEASS

One way to remedy the problem of moderate correlation was proposed by Emiya et al. [34] by the introduction of PEASS measures.

To this end the distortion signal $e(t) \triangleq \hat{x}(t) - x(t)$ is decomposed into target distortion $e^T(t)$, interference $e^I(t)$ and artefacts $e^A(t)$ such that

$$e(t) = e^T(t) + e^I(t) + e^A(t) \qquad \text{(distortion signal)}. \tag{2.72}$$

A perceptually reasonable decomposition is achieved by approximating the auditory time-frequency resolution with the help of a gammatone filterbank and then projecting the distortion signal on a subspace spanned by the source signal. The decomposed errors are then rated according to the perceptual similarity measure (PSM) that was proposed within the development of PEMO-Q [35]. This yields the following four features:

$$q^O \triangleq PSM(\hat{x}(t), x(t)) \qquad \text{(overall perceptual similarity)}, \tag{2.73}$$
$$q^T \triangleq PSM(\hat{x}(t), \hat{x}(t) - e^T(t)), \tag{2.74}$$
$$q^I \triangleq PSM(\hat{x}(t), \hat{x}(t) - e^I(t)), \tag{2.75}$$
$$q^A \triangleq PSM(\hat{x}(t), \hat{x}(t) - e^A(t)). \tag{2.76}$$

These features are then combined into four objective measures with the use of a non-linear mapping function implemented as an artificial neural network (ANN):

- The Overall Perceptual Score (OPS), which reflects the perceived global quality
- The Target-related Perceptual Score (TPS), which measures how much of the target source is preserved
- The Interference-related Perceptual Score (IPS), which measures how much of the other sources is suppressed
- The Artefacts-related Perceptual Score (APS), which quantifies additionally introduced artefacts.

Every measure ranges from 0 to 100 (best). The ANN was trained in a supervised fashion to minimize the error between the proposed objective measures and the subjective evaluation scores obtained in preceding listening tests.

2.5.2 Intelligibility measures

The contributing factors to speech intelligibility are not yet fully understood but most intelligibility measures make use of the assumption that intelligibility mainly depends on the audibility of the target signal in each time-frequency bin (TF-bin) [1, pp. 510f]. Audibility is often measured in terms of the SNR where TF-bands with positive SNR are more likely to contribute to intelligibility than TF-bands with a negative SNR. Thus, the majority of intelligibility measures can be written in the form of

$$\sum_{n,k} W(n,k) \cdot \mathrm{SNR}(n,k) \qquad \text{(speech intelligibility score)}, \tag{2.77}$$

where $W(n,k)$ represents the weights placed on each frequency band and is called the band importance function.Different intelligibility measures have been proposed, varying in the band importance function and the method of SNR calculation. E.g. there are measures based on the Articulation Index (AI measures), others are based on the Speech Transmission Index (STI) and the Modulation Transfer Function (MTF), or there exist also coherence-based measures (see [1, pp. 516ff] for a detailed survey).

In a study conducted in [1, pp. 566] it was also shown that PESQ scores provide a moderately good correlation with speech intelligibility. Furthermore, also the HITFA-score (see 2.5.3) can be considered as an intelligibility measure.

2.5.3 Evaluation of masks

The accuracy of an estimated binary masks BM(n,k) is usually measured against the IBM, where two possible errors can occur: Miss Errors (ME) where the TF-bin is classified as noisy but is actually speech (SNR $>$ 0) and False Alarms (FA) where the TF-bin is classified as speech but is actually noise (SNR $<$ 0). According to [1, pp. 646f], FA errors decrease speech intelligibility more than ME errors, but nevertheless a simple symmetric measure based on the HIT probability (=1-ME) and the FA probability was proposed

$$\mathrm{HITFA_{BM}} = \mathrm{HIT} - \mathrm{FA}, \tag{2.78}$$

which was found to correlate well with speech intelligibility [1, pp. 647].

The error of a softmask SM can be calculated by the MSE between the estimated and the ideal softmask (ISM)

$$\mathrm{MSE_{SM}} = \mathbb{E}[(\mathrm{SM}(n,k) - \mathrm{ISM}(n,k))^2], \tag{2.79}$$

although it is primarily used as an objective function for supervised learning algorithms.

2.6 Challenges and problems

SE and SCSS approaches face various problems, some that are inherent to the domain and some that are specific to certain algorithms or the whole research field. The following list gives a short overview:

- *Masking*: One major problem in the domain of SE is the problem of masking. Many past algorithms employ some sort of noise-estimate which serves as a basis for the decision of how much attenuation is applied to one time-frequency bin. With decreasing SNR the additive contribution of low energy speech components to the noise floor becomes very small, usually much smaller than the variance of the noise-estimate. Thus, the attenuation applied to low-energy speech components will be similar as for time-frequency bins that only contain noise and therefore the low-energy speech components are lost. In more recent work, this problem is partly mitigated by employing speech and noise models (compare [4, 7]) but this approach leads to complex and often highly specialized models (e.g. [24]).

- *Noise statistics*: To further complicate the issue, learning suitable noise statistics seems to be a hard problem. Where in past studies, noise was primarily treated as stationary or slowly changing, current research has a strong focus on highly non-stationary noise types. Contrary to speech, that has a low intrinsic dimensionality because its production is constraint by how our vocal tract works, noise can have a much higher variability. In this sense, noise is every possible sound except the target, which makes it inherently harder to learn adequate models.

- *Introduced artefacts*: In case of a slightly wrong noise-estimate, incorrect heavy attenuation might be applied in one frame but not in its successor. This generates energy peaks and valleys which randomly change from frame to frame, yielding a very tonal form of

noise called musical noise. Although this has mainly been a problem of spectral subtraction algorithms, binary masks face the same problem even to a bigger extend as their suppression function is highly unsmooth.

- *Learned representations*: Many state-of-the-art machine-learning algorithms (deliberately) do not incorporate much domain knowledge, but are rather trained on raw representations as e.g. spectrograms. Although this might be favourable for generative learning, extracting meaningful features of an unconstrained input-space is a hard problem and imperfect representations can also easily yield artefacts that violate basic speech characteristics.

- *Psychoacoustic sensitivity*: The latter problem is also indicative for what could be called the *psychoacoustic sensitivity* of a spectrogram. The human perception of speech or sounds in general is to some extend very insensitive to distortions or modifications of the spectrogram - a fact that is often exploited by audio-compression algorithms [36]. This however is only true for distortions that are psychoacousticly irrelevant. Other spectral modifications lead to clearly audible distortions resulting in a loss of quality of the processed signal. In other words, the impact of small, not psychoacoustic-aware changes in the spectrogram can be very high, yielding perceptually important changes in the processed signal.

- *Evaluation*: Many objective measures have been proposed and although some show moderately high correlation with subjective listening tests, objective evaluation is still an open problem. To our current knowledge, no score has proven to work reliably over different SNRs and all SE/SCSS-settings (compare [1, pp. 514]).

- *Comparability*: Another issue is that although standardized noise corpora and measures exist, many studies use different settings and metrics which makes them often hardly comparable.

3 Sum-Product networks (SPNs)

The intention of this chapter is to review the fundamentals of SPNs and provide some additional details on the training-algorithm.

SPNs as defined in [37] are rooted, directed acyclic graphs (DAGs) which contain sum-, product- and leaf-nodes. Although they are strictly more general, the can be used to compactly represent a network polynomial (NP). NPs were originally defined in [38] as a polynomial representation of the probabilities of a Bayesian network with binary variables. Probabilistic queries such as inference (computing the posterior probabilities of not observed random variables (RVs), given some observation) can be answered by evaluating and differentiating this polynomial. It is exponential in the number of nodes of the Bayesian network and therefore not representable for bigger networks. Despite that, there might be compact representations of it, which adhere to the same probabilistic semantics but also make inference tractable.

3.1 Network Polynomials

For every Bayesian network with binary variables $\boldsymbol{X}$ it is possible to define a multilinear function composed by evidence indicators (λ_X) and network parameters (θ_X), where each term represents one possible state of $\boldsymbol{X}$. In the case of the Bayesian network $Z \to Y$, the NP is given by

$$f(\lambda_y, \lambda_{\bar{y}}, \lambda_z, \lambda_{\bar{z}}) = \theta_{y|z}\theta_z\ \lambda_y\lambda_z + \theta_{\bar{y}|z}\theta_z\ \lambda_{\bar{y}}\lambda_z + \theta_{y|\bar{z}}\theta_{\bar{z}}\ \lambda_y\lambda_{\bar{z}} + \theta_{\bar{y}|\bar{z}}\theta_{\bar{z}}\ \lambda_{\bar{y}}\lambda_{\bar{z}}, \tag{3.1}$$

where the bar over the value of an RV means, that the variable appears negated. More generally, for a Bayesian network over the variables $\boldsymbol{X}$, where $\boldsymbol{U}$ denotes the parents of one node X, its NP is defined as

$$f = \sum_{\boldsymbol{x} \in \boldsymbol{X}} \prod_{x\boldsymbol{u} \sim \boldsymbol{x}} \lambda_x\ \theta_{x|\boldsymbol{u}}. \tag{3.2}$$

The outer sum ranges over all possible instantiations $\boldsymbol{x}$ of $\boldsymbol{X}$, the inner product ranges over all $x\boldsymbol{u}$ that are compatible with the instantiation $\boldsymbol{x}$.

Given that some variables $\boldsymbol{E} \subseteq \boldsymbol{X}$ are observed with the values $\boldsymbol{e}$, we can compute the probability of this evidence $\mathrm{P}(\boldsymbol{e})$ by setting every evidence indicator λ_X that is consistent with

$\boldsymbol{e}$ to 1 and all others to 0. In the previous example under full evidence $\boldsymbol{e} = \bar{y}z$ this gives

$$\mathrm{P}(\boldsymbol{e}) = f(\boldsymbol{e}) = \mathrm{P}(Y{=}\bar{y}, Z{=}z) = f(0,1,1,0) = \theta_{\bar{y}|z}\theta_z, \tag{3.3}$$

which is in correspondence to how one would calculate the probability of this exact state given the Bayesian network. For partial evidence $\boldsymbol{e} = \bar{y}$

$$\mathrm{P}(Y{=}\bar{y}) = f(0,1,1,1) = \theta_{\bar{y}|z}\theta_z + \theta_{\bar{y}|\bar{z}}\theta_{\bar{z}} \tag{3.4}$$

it can be seen that this corresponds to summing out the variable Z. In fact, by this definition *every* $X \notin \boldsymbol{E}$ will by summed out yielding the marginal *probability* of evidence.

The partial derivative of the NP with respect to one evidence indicator corresponds to conditioning the polynomial (not the probability distribution) on exactly this state. For the previous example the derivation with respect to $\lambda_{\bar{y}}$ gives the *distribution*

$$\partial f/\partial\lambda_{\bar{y}} = \mathrm{P}(Y{=}\bar{y}, Z) = \theta_{\bar{y}|z}\theta_z\ \lambda_z + \theta_{\bar{y}|\bar{z}}\theta_{\bar{z}}\ \lambda_{\bar{z}}. \tag{3.5}$$

For a given evidence $\boldsymbol{e}$, the derivative with respect to some evidence indicator λ_x yields the probability

$$\left[\frac{\partial f}{\partial \lambda_x}\right](\boldsymbol{e}) = \mathrm{P}(X{=}x, \boldsymbol{E}{=}\boldsymbol{e}{-}X), \tag{3.6}$$

where $\boldsymbol{e}-X$ denotes all instantiations $\boldsymbol{e}$ reduced by the instantiations of X. To normalize this quantity with respect to the evidence we can make use of the product rule of probability

$$\mathrm{P}(X{=}x \,|\, \boldsymbol{E}{=}\boldsymbol{e}) = \frac{\mathrm{P}(X{=}x, \boldsymbol{E}{=}\boldsymbol{e})}{\mathrm{P}(\boldsymbol{E}{=}\boldsymbol{e})} = \frac{1}{f(\boldsymbol{e})}\left[\frac{\partial f}{\partial \lambda_x}\right](\boldsymbol{e}),\ \ X \notin \boldsymbol{E}. \tag{3.7}$$

This means that by calculating all derivatives of the NP with respect to the evidence indicators, the posterior marginals given the evidence are efficiently computable if the NP can be evaluated efficiently.

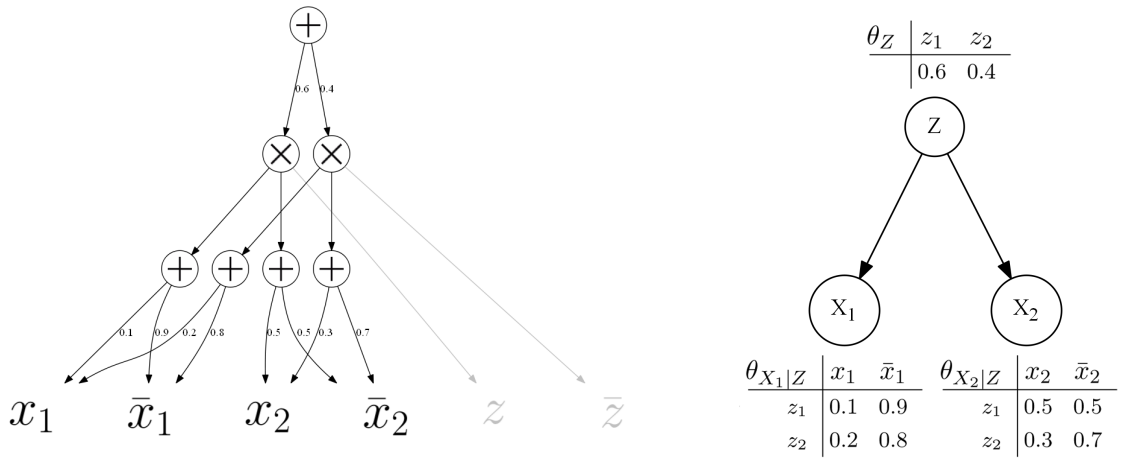
3.2 Foundations of SPNs

One way of efficiently representing and evaluating a NP is an SPN. While Darwiche [38] only considered NPs as a representation of a Bayesian network, they can easily be extended to unnormalized distributions, replacing normalized (conditional) distributions (denoted θ_X in Section 3.1) by unnormalized factors $\Phi(X)$. Loosely adopting the notation of [37], for an unnormalized probability distribution $\Phi(\boldsymbol{X})$ its network polynomial is given by

$$f = \sum_{\boldsymbol{x}\in\boldsymbol{X}} \Phi(\boldsymbol{x})\Pi(\boldsymbol{x}), \tag{3.8}$$

where $\boldsymbol{x}$ ranges over all possible instantiations of $\boldsymbol{X}$ and $\Pi(\boldsymbol{x})$ is the monomial defined as the product of indicators that are 1 in state $\boldsymbol{x}$ (indicators will be denoted as $x_i = [X_i = 1]$ and $\bar{x}_i = [X_i = 0]$). The (unnormalized) probability of evidence $\Phi(\boldsymbol{e})$ is calculated as in Section 3.1 and it can be normalized by the partition function $\mathcal{Z}$ which is obtained by to setting all network-indicators to 1.

An SPN over the variables $\boldsymbol{X}$ is defined as a rooted DAG that contains layers of sum-nodes, product-nodes and indicator leaf nodes. W.l.o.g. the sum- and product layers can be assumed to be alternating. The value of a product node is given by the product of the values of its child nodes $\prod_{j\in\mathrm{ch}(i)} v_j$, the value of a sum-node is a weighted sum of the values of its child nodes



(a) Example SPN (b) Corresponding Bayesian network

Figure 3.1: An example SPN. Without considering the indicators z and $\bar{z}$, the SPN can be interpreted as summing out a latent variable Z corresponding to the root node. When the indicators of Z are observed, the SPN can be interpreted as a naive Bayes model classification, where z and $\bar{z}$ represent the two possible classes.

$\sum_{j\in\mathrm{ch}(i)} w_{ij}v_j$, where $\mathrm{ch}(i)$ denotes the children of node i, v_j the value of node j and w_{ij} the non-negative weight of the edge (i,j). The value of the SPN is the value of its root node and similar as the network polynomial it is a function over its indicator variables. A complete state $\boldsymbol{x}$ of the SPN S is denoted $S(\boldsymbol{x})$, partial evidence is written as $S(\boldsymbol{e})$. The scope of a node $\mathrm{sc}(\cdot)$ is the subset of random variables that appear in it.

An SPN defines an unnormalized probability distribution for all $\boldsymbol{x} \in \boldsymbol{X}$. The unnormalized probability of evidence $\Phi_S(\boldsymbol{e})$ is the sum of all states $\boldsymbol{x}$ that are consistent with $\boldsymbol{e}$, the partition function $\mathcal{Z}_S$ is the sum over all possible states $\boldsymbol{x}$, i.e.

$$\Phi_S(\boldsymbol{e}) = \sum_{\boldsymbol{x}\sim\boldsymbol{e}} S(\boldsymbol{x}), \quad \mathcal{Z}_S = \sum_{\boldsymbol{x}\in\boldsymbol{X}} S(\boldsymbol{x}). \tag{3.9}$$

Figure 3.1(a) provides an example SPN. If we for now ignore the indicators z and $\bar{z}$ – the computation performed by the network can then be written as

$$\begin{aligned} S(x_1,\bar{x}_1,x_2,\bar{x}_2) = {} & 0.6\,[(0.1x_1 + 0.9\bar{x}_1)(0.5x_2 + 0.5\bar{x}_2)] \\ & + 0.4\,[(0.2x_1 + 0.8\bar{x}_1)(0.3x_2 + 0.7\bar{x}_2)] \end{aligned} \tag{3.10}$$

and the network polynomial takes the form

$$\begin{aligned} f(x_1,\bar{x}_1,x_2,\bar{x}_2) = {} & (0.6\cdot 0.1\cdot 0.5 + 0.4\cdot 0.2\cdot 0.3)x_1x_2 \\ & + (0.6\cdot 0.9\cdot 0.5 + 0.4\cdot 0.8\cdot 0.3)\bar{x}_1x_2 \\ & + (0.6\cdot 0.1\cdot 0.5 + 0.4\cdot 0.2\cdot 0.7)x_1\bar{x}_2 \\ & + (0.6\cdot 0.9\cdot 0.5 + 0.4\cdot 0.8\cdot 0.7)\bar{x}_1\bar{x}_2. \end{aligned} \tag{3.11}$$

The equivalent Bayesian network is given in Figure 3.1(b). Without the indicators z and $\bar{z}$, the network can be seen as having 2 observed variables X_1 and X_2 and one latent (unobserved) variable Z, corresponding to the root sum-node. The latent variable has the marginal distribution $\mathrm{P}(Z)$ given by the edges of the root node, which act as a prior distribution over its features, i.e. the child product nodes. Each product node represents one conditional distribution over X_1 and X_2, given one particular state z of the latent variable ($\mathrm{P}(X_1, X_2 \mid z)$), each child sum-

node represents the distribution over one variable given the latent state ($\mathrm{P}(X_i \mid z)$). In terms of probabilities the example SPN of Figure 3.1(b) can be written as

$$\begin{aligned} P(X_1, X_2) &= \mathrm{P}(z_1)\left[\big(\mathrm{P}(x_1|z_1) + \mathrm{P}(\bar{x}_1|z_1)\big)\big(\mathrm{P}(x_2|z_1) + \mathrm{P}(\bar{x}_2|z_1)\big)\right] \\ &\quad + \mathrm{P}(z_2)\left[\big(\mathrm{P}(x_1|z_2) + \mathrm{P}(\bar{x}_1|z_2)\big)\big(\mathrm{P}(x_2|z_2) + \mathrm{P}(\bar{x}_2|z_2)\big)\right] \\ &= \sum_Z \mathrm{P}(Z)\left[\mathrm{P}(X_1 \mid Z)\mathrm{P}(X_2 \mid Z)\right] \\ &= \sum_Z \mathrm{P}(Z)\mathrm{P}(X_1, X_2 \mid Z) \\ &= \sum_Z \mathrm{P}(Z, X_1, X_2). \end{aligned} \tag{3.12}$$

Thus, the sum node can be seen as summing out a hidden variable Z. That intuition can be extended to every sum-node in an SPN: For summing out a variable of a distribution represented by a network polynomial, according to [38], all its indicators have to be set to 1. If an SPN correctly represents its underlying network polynomial and we consider a sum node as a variable of the network, each state of the variable corresponds to one child-node, thus we can think of every product-child having an additional indicator attached. For summing out, each of these virtual indicators are set to one, which is equivalent to the original network.

When considering the variable of Z as observed, the calculation performed by the network is the same as in a naive Bayes model where we want to calculate the likelihood of a specific class Z, given the data $\{X_1, X_2\}$: $\mathrm{P}(Z \mid X_1, X_2) \propto \mathrm{P}(X_1, X_2 \mid Z)\mathrm{P}(Z)$.

The properties of this example however do not hold for every SPN as the weights of sum-nodes, corresponding to distributions of observed or hidden variables might not be normalized and there might also be SPNs that do not correctly represent a network polynomial. Poon and Domingos [37] define an SPN as *valid* iff the network correctly computes the unnormalized probability of evidence, i.e. $S(\boldsymbol{e}) = \Phi_S(\boldsymbol{e})$ for all $\boldsymbol{e}$. That means, that for every (partial) evidence $\boldsymbol{e}$, the value of the network, calculated by setting the indicators of all variables not appearing in $\boldsymbol{e}$ to 1, has to be the same as the sum of all complete states $\boldsymbol{x}$ that are compatible with $\boldsymbol{e}$. From this immediately follows that the partition function of a valid SPN is the value of the SPN with all indicators set to 1, i.e. $\mathcal{Z} = \mathcal{Z}_S$.

Poon and Domingos proof that *validity* follows from 2 general conditions, namely *completeness* and *consistency*. A network is *complete* if all children of a sum-nodes have the same scope. A network is *consistent* if no variable appears negated in one child and non-negated in another child of a product-node.

An intuition of these concepts can be provided by the expansion of an SPN, which is obtained by applying the distributive law bottom-up to all product nodes, treating each leaf x_i as $1x_i + 0\bar{x}_i$ and $\bar{x}_i$ as $0x_i + 1\bar{x}_i$. The expansions $\dot{S}$ of the invalid networks in Figure 3.2 are given by

$$\dot{S}_m = 0.3(1x_1 + 0\bar{x}_1) + 0.7(0x_2 + 1\bar{x}_2) \qquad = 0.3x_1 + 0\bar{x}_1 + 0.7x_2 + 0\bar{x}_2, \tag{3.13}$$

$$\dot{S}_n = (1x_1 + 0\bar{x}_1)(0x_1 + 1\bar{x}_1) \qquad = 0x_1x_1 + 1x_1\bar{x}_1 + 0\bar{x}_1x_1 + 0\bar{x}_1\bar{x}_1, \tag{3.14}$$

their network polynomials follow directly from the joint distribution

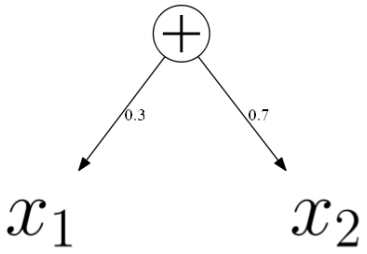
$$f_m = 1x_1x_2 + 0.7x_1\bar{x}_2 + 0.3\bar{x}_1x_2 + 0\bar{x}_1\bar{x}_2, \tag{3.15}$$

$$f_n = 0x_1 + 0\bar{x}_1. \tag{3.16}$$

For evidence $\boldsymbol{e} = x_1$ the incomplete SPN example evaluates to

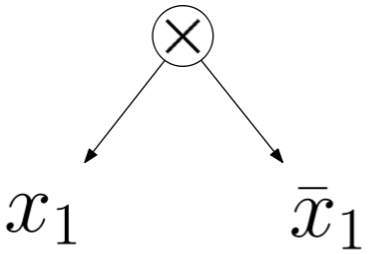
$$S_m(\boldsymbol{e}) = \dot{S}_m(1, 0, 1, 1) = 1, \qquad \Phi_{S_m} = S_m(1, 0, 1, 0) + S_m(1, 0, 0, 1) = 1.3, \tag{3.17}$$

so that it can be seen that the incomplete SPN tends to undercount the summation $\Phi_S =$



(a) Incomplete SPN (S_m)

X_1	X_2	$\Phi_{S_m}(\boldsymbol{X})$
0	0	0
0	1	0.7
1	0	0.3
1	1	1



(b) Inconsistent SPN (S_n)

X_1	$\Phi_{S_n}(\boldsymbol{X})$
0	0
1	0

Figure 3.2: Example minimal incomplete and inconsistent SPNs with their respective probability distributions.

$\sum_{\boldsymbol{x}\sim\boldsymbol{e}} S(\boldsymbol{x})$, i.e. $S(\boldsymbol{e}) \leq \Phi_S$. This is because both probabilities $\mathrm{P}(x_1, x_2) = x_1 + x_2$ and $\mathrm{P}(x_1, \bar{x}_2) = x_1 + \bar{x}_2$ include the factor x_1, which only appears once in the expansion of the network. Some monomials in the expansion of the SPN are therefore missing indicators relative to the monomials in the network polynomial.

For the inconsistent example SPN with evidence $\boldsymbol{e} = \{\}$ (which is equal to calculating $\mathcal{Z}_{S_n}$) we get

$$S_n(\boldsymbol{e}) = \dot{S}_n(1,1) = 1, \qquad \Phi_{S_n} = S_n(1,0) + S_n(0,1) = 0. \tag{3.18}$$

The network tends to overcount the summation $S(\boldsymbol{e}) \geq \Phi_S$ because its expansion includes monomials that are not present in the network polynomial.

From this follows the intuition that an SPN is *valid* if the monomials of its expansion are in one-to-one correspondence with the monomials of its network polynomial, which can be proven to be the case when a network is *complete* and *consistent*. *Completeness* and *consistency* are however not necessary conditions for *validity* (e.g. the network $S = 0.5x_1x_2\bar{x}_2 + 0.5x_1$ is *valid* but *incomplete* and *inconsistent*). *Consistency* can be replaced by a slightly easier but more restrictive concept called *Decomposability*, so that a network is *decomposable* if all child-nodes of a product node have non-overlapping scopes. If an SPN is *complete* and *decomposable* and all weights of every sum-node sum to 1, all probabilistic interpretations of the example SPN above are valid, i.e. that a sum-node corresponds to summing out a hidden variable and that every internal non-root node represents a normalized probability distribution that is conditioned on the states of all latent variables/sum-node between this node and the root node. Product-nodes act as crossovers between distributions of different scopes, assuming mutual independence (i.e. $\mathrm{P}(A, B) = \mathrm{P}(A)\mathrm{P}(B)$ only holds for independent variables A and B). They can be seen as higher-order features of the input variables. Sum-nodes define mixtures over distributions (which are each also products of mixture distributions) dissolving the independence assumptions made by product-nodes (see Figure 3.3 for an example). Each child of a sum-node can be seen as one component of a mixture distribution where the weights function as the components priors. From this recursive viewpoint, an SPN is a compact way the specify a distribution with exponentially many high-level mixture components which are formed over lower-level features, represented itself by mixture distributions.

SPNs can easily be generalized to non-binary distributions by replacing the boolean indicators of the random variable X_i by one indicator $x_i^j = [X_i = \mathrm{val}_i(j)]$ where $\mathrm{val}_i(\cdot)$ is the set of possible

values X_i can take. A multinomial distribution for J different values can then be represented with a sum-node as $\mathrm{P}(X_i) = \sum_{j=1}^{J} \mathrm{P}(X_i = x_i^j) x_i^j$. Furthermore SPNs can be generalized to continuous distributions by replacing the sum of weighted indicators with an integral $\int p(x)dx$ where $p(x)$ is a probability distribution function. Instead of sum-nodes with indicator children, a network over continuous variables then contains distribution nodes, e.g. Gaussian distribution nodes.

3.2.1 Inference

As complete and decomposable SPNs are a representation of a network polynomial[6], all inference methods described in Section 3.1 and [38] can also be performed on the network, i.e. we can calculate the marginals of all (observed and latent) variables by differentiation. For a complete state $\boldsymbol{x}$, let us denote the value of a subnetwork of the SPN S rooted at an arbitrary node n_i, which itself is an SPN, as $S_i(\boldsymbol{x})$ and $\mathrm{pa}(i)$ as the parents if node n_i. First we note that if n_i is the root node $\partial S(\boldsymbol{x})/\partial S_i(\boldsymbol{x}) = 1$, as the value of the SPN is the value of the root. The partial derivatives for inner nodes can then be derived by using the chain rule of differential calculus. If n_i is a product node, all its parents n_k are by definition sum-nodes:

$$\begin{aligned}
\frac{\partial S(\boldsymbol{x})}{\partial S_i(\boldsymbol{x})} &= \sum_{k \in \mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})} \frac{\partial S_k(\boldsymbol{x})}{\partial S_i(\boldsymbol{x})} \\
&= \sum_{k \in \mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})} \frac{\partial \left(\sum_{j \in \mathrm{ch}(k)} w_{ij} S_j(\boldsymbol{x}) \right)}{\partial S_i(\boldsymbol{x})} \\
&= \sum_{k \in \mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})} w_{ki}. \qquad (3.19)
\end{aligned}$$

Note that the indices i and j both denote children of n_k, thus all terms except where $i{=}j$ are cancelled out.

If n_i is a sum node, all its parents n_k are by definition product-nodes, thus

$$\begin{aligned}
\frac{\partial S(\boldsymbol{x})}{\partial S_i(\boldsymbol{x})} &= \sum_{k \in \mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})} \frac{\partial S_k(\boldsymbol{x})}{\partial S_i(\boldsymbol{x})} \\
&= \sum_{k \in \mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})} \frac{\partial \left(\prod_{j \in \mathrm{ch}(k)} S_j(\boldsymbol{x}) \right)}{\partial S_i(\boldsymbol{x})} \\
&= \sum_{k \in \mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})} \prod_{j \in \mathrm{ch}(k),\, j \neq i} S_j(\boldsymbol{x}). \qquad (3.20)
\end{aligned}$$

Through recursive application of Equations 3.19 and 3.20, a partial derivative for every node in an SPN can be calculated. An efficient way to do this is to calculate the value of every node in an upward pass from the leaves to the root and calculate the derivatives in a subsequent downward pass.

For calculating the most probable state of all observable variables $\boldsymbol{X}_i$ given a partial evidence $\boldsymbol{e}$, the MPE (most probable explanation) state $\mathrm{argmax}_{\boldsymbol{X},\boldsymbol{Z}} \mathrm{P}(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{e})$ can be calculated by replacing sums with maximizations and setting all indicators $\boldsymbol{x}_i \notin \boldsymbol{e}$ to 1. In the upward pass

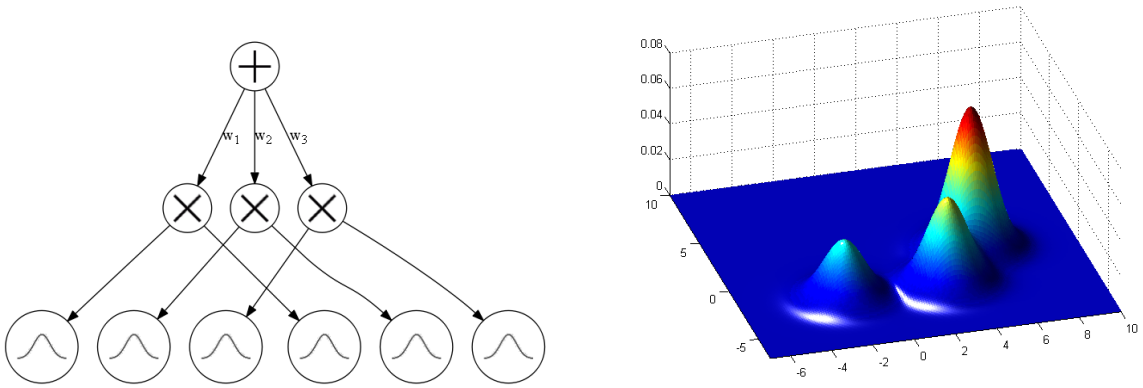
[6] In opposition to [37], this is not true for valid, i.e. complete and consistent SPNs (personal communication with R. Peharz)

the max-nodes output the child with the weighted maximum value, the downward pass then performs a trace-back, calculating the single path which yielded the maxima. It has to be noted that MPE inference calculates the joint most probable state of the observed variables $\boldsymbol{X}$ and the implicit latent variables $\boldsymbol{Z}$. The MAP state of the network $\operatorname{argmax}_{\boldsymbol{X}} \mathrm{P}(\boldsymbol{X}|\boldsymbol{e})$ would require summing over all hidden variables, which is conjectured by Peharz et. al [39] to be intractable.

3.2.2 Learning

On the one hand an SPN can be seen as a deep neural network with weights of the sum-nodes being its parameters. This interpretation gives rise to gradient based learning methods where the likelihood can be optimized by backpropagation making use of Equations 3.19 and 3.20.

On the other hand an SPN can be interpreted as a Bayesian network with a deep structure of latent variables, which opens the door for EM-based learning methods. Figure 3.3 shows an example SPN representing a Gaussian Mixture Model (GMM) with diagonal covariance matrices over 2 dimensions (X_1, X_2) with K=3 mixture components each. For this example we assume that the means of the Gaussians are fixed and that they have unit variance. This is a rather restricted model, because the only thing that can be optimized are the components priors, which corresponds to weights of the sum-nodes in the SPN.



(a) GMM modelled by an SPN

(b) GMM probability distribution

Figure 3.3: Left: SPN modelling a GMM with 3 components over 2 dimensions. Each Gaussian node represents a one-dimensional Gaussian distribution, each product-node a 2 dimensional Gaussian distribution and the sum-node represents the mixture of the three individual Gaussians. All Gaussians have a diagonal covariance matrix as product nodes induce statistical independence between the dimensions. Right: Surface plot of an example SPN/GMM with $\mu = \{-2, -2, 1, -2, 3, 5\}$ *and unit variance,* $w = \{.2, .3, .5\}$.

In the E-Step we fix the parameters, which in this case are the priors of a components w_j, and compute the posteriors of all latent variables $\mathrm{P}(Z \mid \boldsymbol{X})$, which can be seen as the normalized responsibility that one component indexed by j takes for explaining one datapoint $\boldsymbol{x}_n$:

$$\mathrm{P}(Z{=}j \mid \boldsymbol{X}{=}\boldsymbol{x}_n) = \gamma_{nj} = \frac{\mathcal{N}_j(\boldsymbol{x}_n)}{\sum_{l=1}^{K} \mathcal{N}_l(\boldsymbol{x}_n)}, \quad \mathcal{N}_j(\boldsymbol{x}_n) = w_j \, \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j). \tag{3.21}$$

In the M-Step the responsibilities γ_{nj} are kept fixed and the parameters w_i are reestimated. For

GMMs there exists a closed form maximum likelihood estimator which is given by

$$w_j = \frac{N_j}{N}, \quad N_j = \sum_{n=1}^{N} \gamma_{nj}. \tag{3.22}$$

This can be interpreted as setting the prior of the component j proportional to its importance for all datapoints.

As we already mentioned, in this restricted example we do not change the means or the variance of the Gaussians and the optimization is only performed by increasing the importance of the Gaussians that adequately model more datapoints and lowering their importance otherwise.

This variant of EM for GMMs still uses the soft-assignments γ_{nj}. Poon and Domingos [37] state, that although this works in principal, it fails for deep networks because the weight updates get very small when they traverse more layers. Their solution was to use "hard" EM, which uses hard assignments between examples and clusters, much like k-means. The hard assignment is achieved by replacing marginal inference (calculation of $\mathrm{P}(\boldsymbol{Z} \mid \boldsymbol{X})$) by MPE-inference, so that for one example $\boldsymbol{x}_n$ there is only one path (conceptually corresponding to one centroid of k-means) that takes credit for this example. One can think of it as replacing γ_{nj}, which represents the responsibilities of the components as normalized fractions by r_{nj} having a "1-of-K"-representation, so that only 1 of the K elements is 1, and all others are 0. This avoids any diffusion of the learning signal because all updates can be of unit size (see 3.2.3).

Structure of SPNs

For learning the structure of an SPN, Poon and Domingos propose to start with a generic, densely connected SPN that is complete and consistent and process all examples until convergence. Afterwards all edges with zero weights (i.e. features that were not advantageous for explaining correlations in the data) can be pruned yielding a sparse network. This approach however is infeasible for high dimensional data as this would require the creation of an exponential amount of nodes.

For 2 dimensional data satisfying certain neighbourhood constraints (e.g. image data), they propose an algorithm that starts with a root rectangle covering the whole data array and recursively performs all possible decomposition into two sub-rectangles along both dimensions using a defined stepsize . Let $R_s = \{r_1, r_2, \ldots, 1\}$ denote the set of possible stepsizes, where the values r_s are in descending order. The splitting of parent rectangle R is executed along one dimension with the stepsize r_1 if width and height of R are both bigger than r_1. If one dimension is smaller or equal, the next smaller value of R_s is used to perform the same process.

The root rectangle is linked to a single sum-node which represents the overall distribution modelled by the SPN. Every non-root rectangle that contains more than one variable is equipped with δ sum-nodes and every rectangle of size 1 is linked to γ Gaussian probability density nodes, which are the leaves of the SPN. Their mean is set to the γ quantile means of the variable modelled by the rectangle and they have unit variance (note that as the means and variances are fixed a priori, this corresponds exactly to the restrictions imposed on the example SPN in Section 3.2.2).

Finally, given two sub-rectangles R' and R'' resulting from the split of R, for each combination of one node of R' and one node of R'' (sum- or leave-nodes) a product node is generated and connected as a child to each of the sum-nodes corresponding to R.

3.2.3 Implementational details

The EM algorithm used in [40] is a form of online-EM, i.e. the weights are updated after every example. For achieving updates of unit size, the algorithm not only maintains the weight of each edge of a sum-node, but also a count that indicates how often that specific edge has been part of the MPE-path through the network. The weights then result from normalizing the counts for all edges of a sum-node, additionally applying add-one smoothing. This corresponds to modifying Equation 3.22 to

$$w_j = \frac{N_j + 1}{N + K}, \tag{3.23}$$

which yields a distribution that is a little bit closer to the uniform distribution, i.e. it "smooths" the original distribution of the counts.

Although the structure described in 3.2.2 for modelling image data is sparser than a fully connected graph, it is still large if all nodes are pregenerated. Thus sum- and product-nodes are generated on demand where sum-nodes are generated in the upward pass and product-nodes are generated in the downward pass.

The sparseness prior is an l_0 prior, i.e. it penalizes the l_0 norm of the weights $||\boldsymbol{w}||_{l_0} = \sum_i [w_i \neq 0]$. This is achieved by dividing the count increment by a sparseness factor, thus requiring that the MAP product-node needs to have a high likelihood in comparison with the competing product nodes.

3.3 Applications and Extensions

In the first paper of Poon and Domingos [37] (PD) SPNs where applied to the problem of image completion. Specifically, the SPN was trained on image patches of 64×64 pixels showing faces of different persons [41]. The task was then to reconstruct the complete face, given an image where one half (top or left) was covered. SPNs performed very well in comparison with other methods, but reconstruction still looked somehow blocky (see [37], Fig. 5).

Dennis and Ventura (DV) [42] extended the original architecture by learning the structure of an SPN prior to learning its weights. This is done by clustering subsets of variables that have similar magnitudes using k-means clustering. It has to be noted that k-means is not used to cluster data instances but variables, which is a rather unusual application. This clustering generates a region graph, which recursively partitions the set of observable variables into pairs of smaller regions. The region graph is subsequently transformed to an SPN where the weights are learned with the same algorithm as in [37]. Applied to the same face-reconstruction problem, DVs method consistently gives better results in terms of log-likelihood and MSE on the test set and also the visual appearance of the face-images is less blocky.

Both algorithms above contain an explicit or implicit notion of locality. The splitting in Poon and Domingos explicitly models the relation of neighbouring pixels/regions, the data-driven structure of Dennis and Ventura implicitly induces neighbourhood relations because neighbouring pixel RVs tend to have similar magnitudes. It is clear that two RVs can still exhibit high (negative) correlation although their magnitudes are highly dissimilar. Thus, Peharz et al. [39] (PeA) and Gens and Domingos [43] (GD) introduce two ways of learning the network structure based on statistical independence tests, both learning the structure and the weights at the same time. GD adopted a top-down approach which starts from the whole set of observed variables and recursively splits them with a product node, if they are approximately independent or groups similar instances with sum nodes, if no clear independence relations hold. PeA strived for a bottom-up approach which starts from building models over small variable scopes which are then merged together into SPNs with bigger scopes. Bottom-up trained SPNs achieve simi-

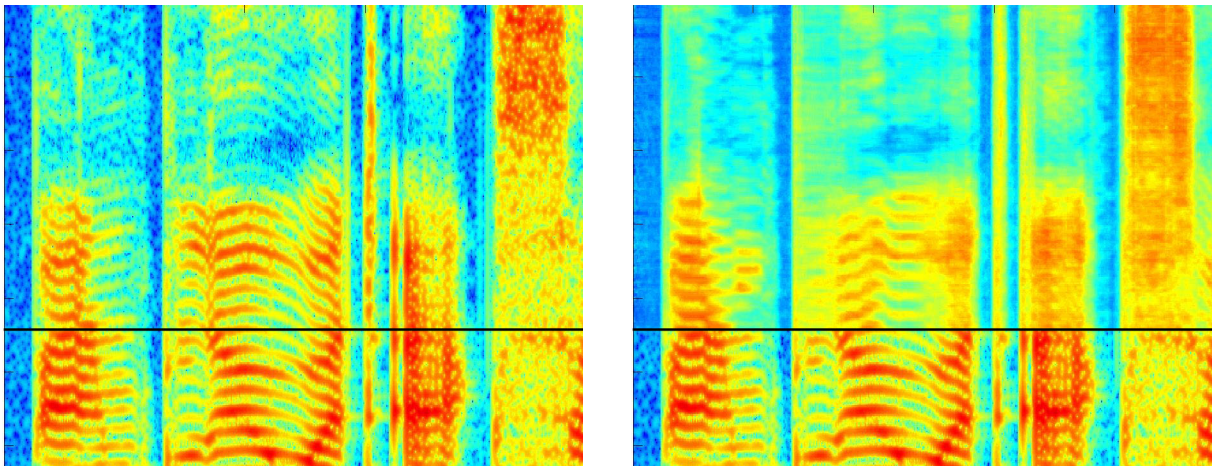
lar performance on the same face-completion task as PD and DV and outperforms them, when their locality assumptions are not met. GD SPNs were evaluated on a wide range of datasets, outperforming standard Bayesian Learners significantly on most of them. No direct comparison between PeA and GD is yet available.

Gens and Domingos [44] showed that SPNs lend themselves also to discriminative learning. To this end, a hard gradient descent method is adopted, where the gradient for backpropagation is not computed by marginal inference but by MPE inference. The proposed architecture was evaluated on two image classification tasks where the highest accuracies up to this date could be achieved.

3.3.1 Artificial Bandwidth extension with SPNs

Another application of SPNs that is particularly interesting in the context of this work is the use of SPNs for artificial bandwidth extension (ABE) [8] of speech signals. Bandwidth extension is the task of re-estimating frequency bands missing in the source signal. For example, even in modern public mobile phone networks, the bandwidth of the transmitted speech is typically limited to a frequency range from approximately 300Hz to 3.4kHz.

The basic idea is to use SPNs for modelling the log-magnitude spectrogram of speech and perform MPE inference in the network with the partial evidence given by the bandpass-filtered signal. The time trajectory of the signal is modelled by an HMM in the way that individual SPNs model the observation probabilities of each state of the HMM. The individual HMM states result from a prior clustering of the data.



(a) Original (b) Reconstruction

Figure 3.4: Artificial bandwidth extension performed with an SPN by reconstructing the frequency bins above the black line where all frequency bins below the black line were given as evidence for the MPE-inference.

As it can be seen in Fig. 3.4 the reconstructions of the spectrograms looks very promising and yielded good results in objective comparison and subjective listening tests. The successful application of SPNs to ABE served as a starting point for this work, as SE/SCSS problems might be similarly representable.

3.4 Extension of this work

For the practical part of this work, the following extensions were applied to the original SPN training and inference algorithms.

Alternative rectangular splitting

As proposed in [37], the structure of an PD SPN is created a priori by recursively splitting a parent rectangle R into all possible decompositions of two subrectangles R' and R'' along both dimensions (see 3.2.2). This splitting happens at different resolutions, depending on the width and height of the parent rectangle. The different resolutions are encoded as stepsizes R_s, which denote the distance the splitting boundary is moved after every splitting. E.g. the splitting of a rectangle with dimension of 6×3 with a stepsize of 2 along the first dimension gives the splittings $2 \times 3|4 \times 3$ and $4 \times 3|2 \times 3$. The splitting of parent rectangle R along one dimension is executed with the biggest stepsize $r_1 \in R_s$ if width and height of R are both bigger than r_1. If one dimension is smaller or equal, the next smaller value of R_s is used to perform the same process.

For data that is not approximately quadratic, this results in splitting the longer edge with a very fine resolution as width *and* height of the rectangle R need to be bigger than the stepsize. This yields a very deep structure of the SPN because as the splitting is applied recursively, this is also true for every subrectangle. E.g. splitting a rectangle with dimension of 6×2 with $R_s = \{3, 1\}$ gives $5 \times 2|1 \times 2$,... where splitting the first subrectangle gives $4 \times 2|1 \times 2$,... etc.

An easy way to fix this is to only consider the size of the dimension the rectangle is split into, i.e. the splitting of parent rectangle R is executed along one dimension with the stepsize r_1 if the length of R in this dimension is bigger than r_1. If it is smaller or equal, the next smaller value of R_s is used for this dimension but not necessarily for the other.

2-channel SPNs

The algorithms by which the structure of a PD SPN is created (see Section 3.2.2) interprets the input variables as being 2-dimensional, where the neighbourhood relations of the variables are defined by the neighbourhood relations of the input array. For layered data where each layer corresponds to a different representation or dimension of the data, we have to adapt this algorithm to account for the different variable interactions.

In the original PD SPNs denoted as 1-channel SPNs, an input variable X_m is allotted $\gamma \in \{1, \ldots, j\}$ 1-dimensional Gaussian probability nodes G_m^j. For modelling 2-channel input data $X_{\{m,n\}}$ we can simply use γ^2 2-dimensional Gaussian probability nodes, which is equivalent to joining each pair of Gaussian probability nodes $\{G_m^j, G_n^j\}$ with a product node.

Examples for this type of data are RGB-images or spectrogram patches with 2 different representation of the same signal.

Constrained MPE-inference (cMPE)

MPE inference can be subject to different constraints in form of soft-bounds on the value-range of every variable. For marginalizing over specific variables X_i in an SPN, we need to integrate over these variables. If an SPN is complete and decomposable, the integral over the root node can be expressed by integrals over its children, which can be recursively propagated through the network. Let S denote a node of an SPN where $\text{sc}(S) = \{X_1, X_2\}$, and assume that we want to marginalize over X_1 and X_2. If S is a sum-node and the network is complete, all of its children

have the same scope, thus by the sum rule of integration we get

$$\int_{X_1,X_2} S(X_1,X_2) = \int_{X_1,X_2} \sum_i w_i S_i(X_1,X_2) = \sum_i w_i \int_{X_1,X_2} S_i(X_1,X_2). \tag{3.24}$$

If S is a product-node and the network is decomposable, all factors have non-overlaping scopes. Thus all nodes of one scope are constant with respect to the integrals of other scopes

$$\int_{X_1,X_2} S(X_1,X_2) = \int_{X_1}\int_{X_2} S_1(X_1)S_2(X_2) = \int_{X_1} S_1(X_1) \int_{X_2} S_2(X_2). \tag{3.25}$$

If S is a Gaussian distribution node of a marginalized variable X_i, we get

$$\int_{X_i} S(X_i) = \int_{-\infty}^{\infty} \mathcal{N}(X_i \,|\, \mu_i^j, \sigma_i^j)\ dx_i = 1, \tag{3.26}$$

where j is the index of the Gaussian node.

Upper bound constraints on the range of X_i can now be thought as giving zero probability to values above a defined threshold, that is, letting the integral only run till the value of the upper bound b_i

$$\int_{-\infty}^{b_i} \mathcal{N}(X_i \,|\, \mu_i^j, \sigma_i^j)\ dx_i = \Phi(b_i \,|\, \mu_i^j, \sigma_i^j), \tag{3.27}$$

where Φ denotes the Gaussian cumulative distribution function.

As MPE inference always returns the mean of the most probable Gaussian probability node, the results is not strictly within the given bounds, i.e. cMPE imposes soft bounds on the reconstructions.

4 Experiments

The starting point of these experiments is the observation we denoted as *masking problem* in 2.6: Especially in low SNR settings, the additive contribution of low-energy speech parts is too weak to be distinguishable from the viewpoint of a noise estimator, that only relies on coarse and speaker-independent spectral statistics. This encourages the use of model-driven approaches, as they can offer very fine-grained speech statistics, that might help to better estimate the low SNR parts of the clean signal.

After some preliminary research (compare Appendix A), two main experiments were conducted: The first experiment tries to directly infer the clean speech (DT-approach) given a subset of time-frequency bins of the mixture, that has a high probability of being speech. The second experiment independently estimates speech and noise from the mixture and combines the two predictions within a soft-mask which is subsequently used to extract the clean speech (IM-approach).

It has to be emphasized that all models presented do not require any feature extraction steps (e.g. mel cepstrum, gammatone filterbank, etc.) but work with the raw FFT data. This is consistent with the generative deep learning paradigm, as it leaves the feature discovery entirely to the network [45].

4.1 Experimental setup

4.1.1 Data setup

As a dataset we used the 2nd CHiME speech separation challenge database [46] consisting of 34 speakers with 500 training samples each with clean and reverb speech signals. We evaluated the performance of our models on 4 different tasks:

- Task SDc: speaker-dependent SE on clean speech data
- Task SDr: speaker-dependent SE on reverberant speech data
- Task SIc: speaker-independent SE on clean speech data
- Task SIr: speaker-independent SE on reverberant speech data

The time-frequency representation $S^{\mathcal{C}}$ was computed by a 512 point Fourier Transform with a hamming window of 32ms length and a stepsize of 10ms. For the generation of the reference masks (IBM and ISM) the isolated noise of the mixture is needed, which is not provided by the test- and evaluation-set data of the CHiME database. Therefore we had to split the training-data into disjoint subsets, where a maximum number of 400 utterances[7] was used for the training-set and 50 utterances each for the test- and validation-set (5 of each speaker for SIc/SIr-tasks).

For the SDc/SDr-tasks, separate models were trained on 1 male and 1 female speaker (speaker IDs {1,11}). For the SIc/SIr-tasks, the training-set consisted of utterances across 5 male and 5 female speakers (speaker IDs {1,2,3,5,6,4,7,11,15,16}). To accommodate for different noise levels, the training-data was mixed at different SNRs {+0dB, +3dB, +6dB}. The evaluation of the models was executed at each of the training SNR levels individually, where for resynthesis of the time-domain signal $\hat{x}(t)$ the noisy phase $\phi_y(n,k)$ was used. All model were trained on normalized log-magnitude spectrograms $log(S(n,k))$, where the normalization was performed for each data-type $\in$ {speech, mixed, noise} separately.

4.1.2 Model setup

For clarity we establish the following terminology: To general SPNs we refer as SPNs, to PD SPNs that were trained on one data-type (speech) we refer as 1-channel SPNs, SPNs that were trained on 2 data-types (mixed and speech or mixed and noise) are named 2-channel SPNs. Either the network is trained on all data samples, yielding 1-channel or 2-channel frame-wise SPNs, or on a subset of data-samples (in conjunction with an HMM), were we denote the networks as 1-channel or 2-channel SPNHMM.

For specifying a specific model configuration, we use the parameters as given in Table 4.1

ID	**Model**	**Type**	**Size**	$\boldsymbol{C}$	$\boldsymbol{W}$, $\boldsymbol{H_T}$	$\boldsymbol{\gamma}$	$\boldsymbol{\delta}$

Table 4.1: Available model parameters.

where *ID* is an sequential identifier, *Size* denotes the number of utterances used for training the model, *Type* $\in$ {S, M2S, M2N} corresponding 1-channel SPNs trained on speech and 2-channel SPNs trained on mixed and speech or mixed and noise respectively, C is the number of states in case of an SPNHMM, W and H_T are defined in the following subsection, and γ and δ are given in Section 3.2.2. For referring to different models, we write the model-ID in angle brackets, e.g. <1> refers to the model with ID 1.

Frame-wise SPNs

SPNs do not posess any inherent capabilities of processing time-varying signals, therefore an input spectrograms $S(n,k)$ of size $N \times K$ is split into smaller spectrogram patches $S^i(w,k)$ of size $W \times K$ by a rectangular window sliding over $S(n,k)$ with the hop size H. If $H < W$ the resulting slices $S^i(w,k)$ contain overlapping spectrogram data. H can obtain different values for training and reconstruction which are denoted by H_T and H_R respectively. Decreasing H_T leads to more (overlapping) training data.

The reconstruction is performed by running MPE inference on patches of an algorithm-dependent mask $\check{S}^i(n,k)$ that contains the observed variables (e.g. Figure 4.2(c)). For assembling the target spectrogram $\hat{S}(n,k)$ out of the separately reconstructed patches $\hat{S}^i(w,k)$, care has to be taken about correct normalization in case of different reconstructions of the same spectrogram

[7] The effective number of training-utterances used was usually less than 400 and is given in the description of each specific experiment.

frame. The reconstruction is given by

$$S(n,k) = \frac{1}{|\mathcal{B}|} \sum \mathcal{B}, \quad \mathcal{B} = \{S^i(w,k) \mid iH_R + w = n\}, \tag{4.1}$$

where $w \in \{0,\dots,W-1\}$ and $i \in \{0,\dots,I-1\}$. For all frame-wise SPN reconstructions we used $H_R = 1$, as this yielded the smoothest estimates.

The general idea of frame-wise SPN models is that with $W > 1$ they can easily incorporate contextual information of adjacent frames that might contain cues about the clean speech signal that are less masked then the ones available for a model with $W = 1$.

SPNHMMs

For SPNHMMs [8], only patches of width $W = 1$ were used, thus splitting and joining the patches is trivial. The HMM is defined by the prior probabilities π, the transition probabilities A and the emission/observation probabilities B. The C individual HMM states result from a prior clustering of the training data, using the LBG clustering algorithm [47]. The clustering is performed on unnormalized data, as this emphasizes the perceptual importance of low frequency bands. Paramters π and A are then estimated by counting the number of patches per state and number of transitions of one state to the next between two successive frames with subsequent normalization. The observation probabilities are modelled by C individual SPNs (denoted as subSPNs), which were trained on the patches corresponding to one region of the clustering.

For reconstruction, each patch is applied an algorithm-specific mask $\check{S}^n(k)$ and inference is run over all patches and models, yielding $\tilde{\mathrm{P}}(S^n(k) \mid C)$, which are the log-likelihoods of each patch given one hidden state/model. These likelihoods serve as input probabilities for the Backward-Forward algorithm, which gives the marginals $\tilde{\mathrm{P}}(C \mid \{S^0(k),\dots,S^n(k)\})$. The reconstruction is then given as a weighted sum of all individual reconstructions

$$S(n,k) = \sum_{c \in C} \tilde{\mathrm{P}}\big(c \mid \boldsymbol{e}^n(k)\big) \cdot \hat{S}_c^n(k), \tag{4.2}$$

where $\hat{S}_c^n(k)$ is the MPE reconstruction of $\check{S}^n(k)$ of the model with index c.

cMPE

The idea of cMPE (Section 3.4) is to bias the reconstruction of the network towards a certain constraint, which for SE is given by the upper bound $\hat{X} \leq Y$. This is fundamentally different than enforcing this constraint in a postprocessing step, as by constraining MPE inference, the network itself tries to comply with the given bounds. Nevertheless, the output of the cMPE (denoted as raw output) is not necessarily strictly within these bounds – for computing the final estimate $\hat{X}$, the constraint is applied subsequently on the raw network output.

4.1.3 Evaluation

Objective evaluation

As objective measures we report PESQ and PEASS scores (compare Section 2.5.1), for comparison of the masks we report the HITFA_{BM} and MSE_{SM} (compare Section 2.5.3) in Appendix C.

Two baselines were generated for every task and are shown in all summary figures in this section:

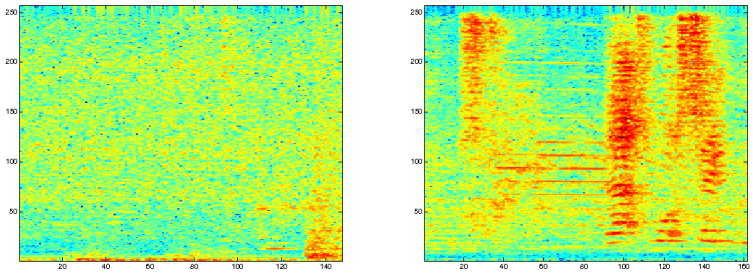
- *Noisy truth*: As speech enhancement can not only improve but also hurt intelligibility and perceptual quality of a signal, a good baseline is given by the mixture itself
- *IMCRA*: To obtain a noise estimate we use IMCRA with standard parameters as given in [17]. The noise estimate is then used within the the MMSE-LSA [18] which gives a denoised version of the mixed signal.

The ground truth, given in Experiment 1 by the clean speech and in Experiment 2 by the speech estimate computed with the ISM or IBM, was also calculated and consistently gives optimal scores (PESQ 4.5, OPS 98.89[8]). Furthermore, for SDr-tasks we compare to the results given in [6, 10]. Figures and results in this chapter are only given for the first task (SDc), more detailed results on all tasks can be found in Appendices B and C.

Subjective evaluation

For subjective visual evaluation, results of every model are discussed on the basis of two examples utterances with different noise types:

- *Easy SE setting*: Utterance-ID *srbu8p* mixed with stationary background noise (Figure 4.1(a))
- *Difficult SE setting*: Utterance-ID *srwi3s* mixed with noise sampled from a living room, including a television and a crying baby (Figure 4.1(b))



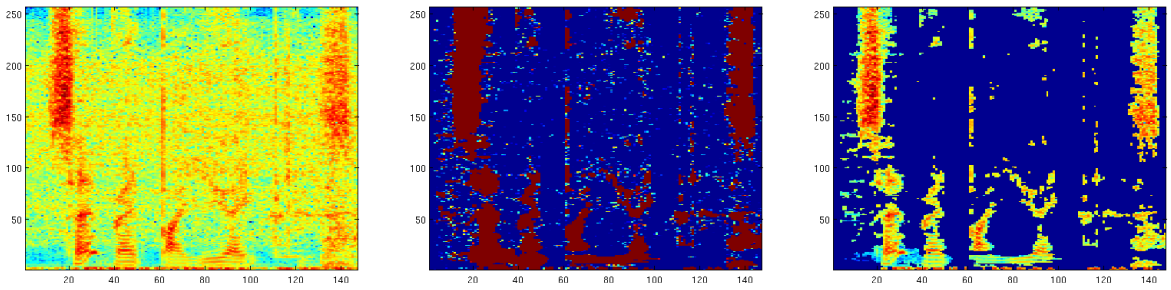
(a) Easy noise (b) Difficult noise

Figure 4.1: Noise in different SE example settings. The utterance mixed with (a) is denoted as easy *SE setting (Task: SDc, noise at +0dB) and the utterance mixed with (b) as* difficult *SE setting (Task: SDc, noise at +0dB).*

[8] Although ISR, SDR, SAR and SIR always gives $+\infty$, PEASS optimal scores are slightly less than 100 due to the learned non-linear mapping (see Section 2.5.1).

4.2 Experiment 1: Direct target estimation with 1-channel SPNHMMs

In [8] we showed that SPNs are well suited for ABE by running MPE inference with the bins of the lowpass-filtered signal are given as evidence. SE can be seen as a similar problem where the evidence is not contained in some known frequency bands, but lies within a subset of TF-bins that are likely to correspond to speech. That is, given some clean speech TF-bins, we can use MPE inference of a 1-channel SPN to predict the missing bins (DT-approach). To obtain the evidence-bins we use the speech presence probabilities of the IMCRA noise estimator (Section 2.3.2) to generate a mask: Bins that according to IMCRA have a high likelihood of being speech are retained and bins that are likely to be noise are discarded (see Figure 4.2).



(a) Noisy mixture (b) Speech presence probabilites $\hat{p}$ (c) Mask

Figure 4.2: IMCRA speech presence probabilities and mask that is used for MPE-reconstruction.

Two factors are crucial in this respect: First, the models must have learned a correct representation of clean speech spectrograms. Second, the cues provided by the noise estimator have to contain enough information for the network to reconstruct the original clean speech signal. This evidently introduces a trade-off between reconstruction accuracy and noise attenuation: The more evidence that is given the network, the more information it has to reconstruct the missing bins, but the more noise will be retained in the estimated clean speech.

Figure 4.3 gives an overview of the objective results obtained in this experiment, a detailed discussion and example plots are given in the following sections.

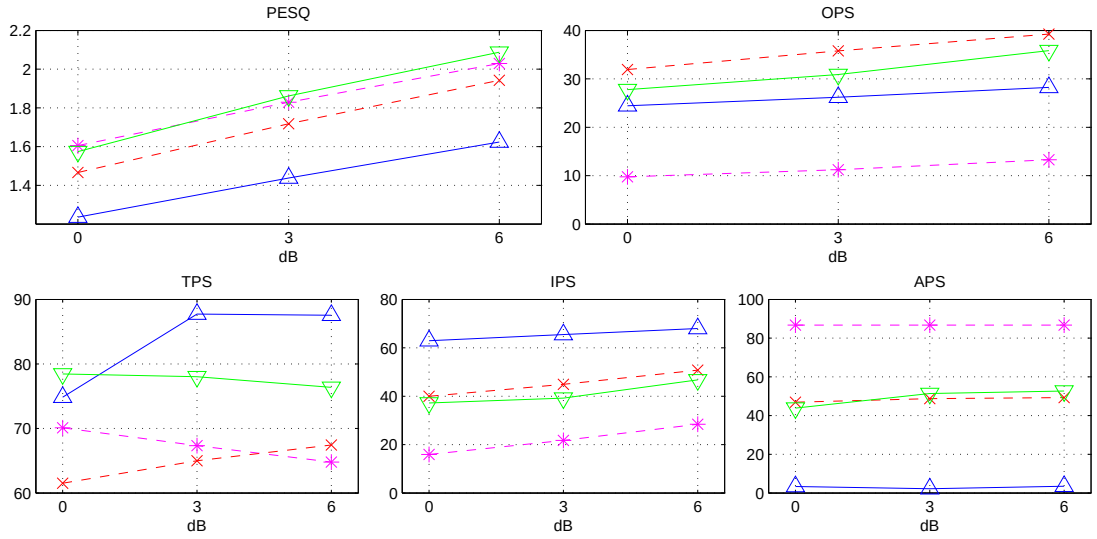


Figure 4.3: Experiment 1 (DT): Results overview for 1-channel SPNHMMs, Symbols: (∗) Noisy truth (×) IMCRA reference (△) SPNHMM (Section 4.2.1) (▽) SPNHMM with cMPE (Section 4.2.2).

4.2.1 Direct speech estimation with 1-channel SPNHMMs

The model used in this section is defined by Table 4.2

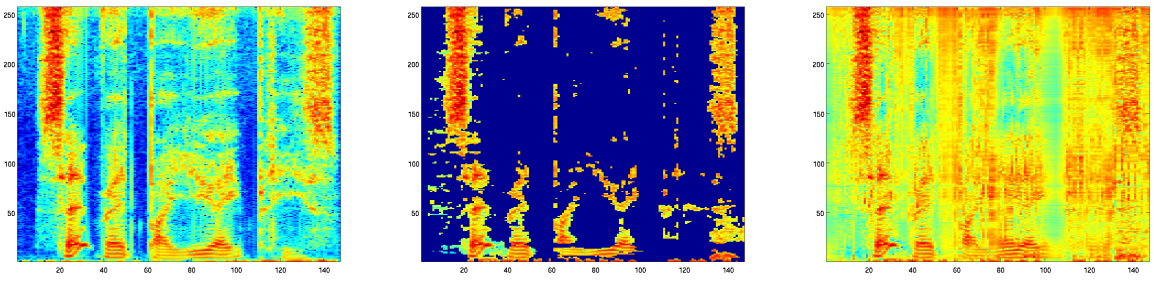
ID	Model	Type	Size	C	W, H	γ	δ
<1>	1ch SPNHMM	S	400	64	1,1	20	20

Table 4.2: Model parameters of the 1-channel SPNHMM used in Experiment 1.

Experiments with 1-channel SPNHMMs showed, that their reconstruction performance is heavily dependent on the speech probability mask provided by IMCRA. Whenever the noise estimate is close to the true noise, the mask most likely contains many true speech TF-bins. However, 1-channel SPNHMMs fail to reproduce a spectrogram representation that is similar to the clean speech data (Figure 4.4(c)). That is also reflected in the PESQ score shown in Figure 4.3. Most notably, it fails to reconstruct areas where is has no or very few speech TF-bins available. The reason for this is that for these frames the HMM often remains in the same states and the results of the MPE reconstruction do not vary much between the subSPNs, given no or few input TF-bins.

The PEASS scores given in Figure 4.3 are consistent with these observations: Although it can increase the overall quality (OPS) with respect to the noisy truth, it has very low APS scores, indicating many additionally introduced artefacts. In terms of interference suppression (IPS), the SPNHMM outperforms both baselines by a large margin and most notably has high TPS values, which might indicate that to some extend masked clean speech bins could be reconstructed. However, it seems more plausible, that this a mere result of predicting high-energy over the whole spectrogram. Subjective listening test were in line with the scores, although the high level of artefact makes 1-channel SPNHMM-enhanced speech very unpleasant to listen to.

It is evident that the estimates can be made more plausible by introducing a postprocessing step, requiring the reconstructions to be strictly less or equal to the magnitude of the mixed spectrograms. Although this improves performance, the postprocessed spectrograms are almost equal to the original noisy input, rendering the reconstructions useless. All results in this section are thus given for 1-channel SPNHMM outputs without post-processing.



(a) Target speech spectrogram (b) Mask (c) Estimate

Figure 4.4: 1-channel SPNHMM with MPE inference (easy SE setting), Model <1> (full example given in Figure B.1).

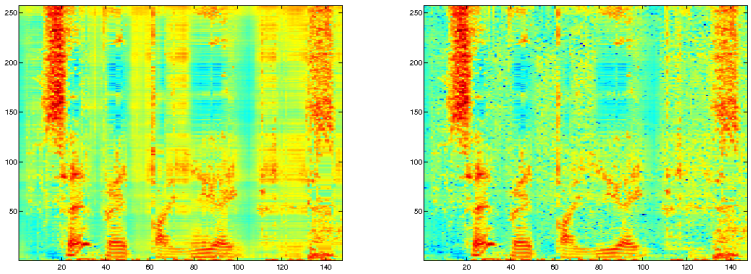
4.2.2 Direct speech estimation with 1-channel SPNHMMs and cMPE

Analysing the speech estimates of model <1> with cMPE inference, we observed that the raw network outputs (cMPE reconstructions without explicitly enforced contraints, see Section 3.4) already exhibit a much higher degree of similarity to the clean speech spectrograms (compare Figure 4.5(a)). Strictly enforcing the constraints further improves their similarity (Figure 4.5(b)).

Nonetheless, this approach equivalently severely suffers from bad noise estimates and in more difficult SE settings (as e.g. in Figure 4.6), IMCRA fails to exclude many TF-bins corresponding to the interfering signal. As the 1-channel SPNHMM reconstructions are bounded by the quality of the noise estimate, the subsequent clean speech estimations will also be poor.

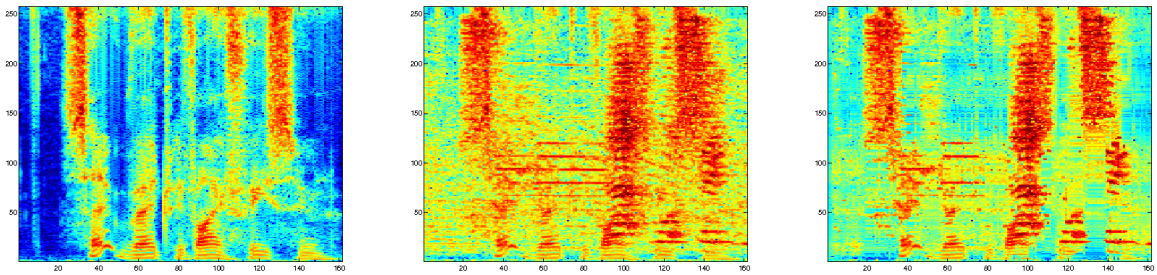
Furthermore, despite the constraint reconstruction, the raw network outputs (e.g. Figure 4.5(a)) still have a higher mean energy than the mixture. This happens when for the subSPNs, the distribution of the training- and reconstruction-data differs in such a way, that for specific parts there are no low-energy Gaussian probability nodes available for fulfilling the constraint.

In terms of PESQ scores (see Figure 4.3), SPNHMMs with cMPE achieve an average relative improvement of 25% over unconstraint SPNHMM reconstructions and do not decrease the quality with respect to the noisy truth. In comparison with IMCRA, no consistent result is achieved as PESQ predicts higher preceptual quality but OPS scores indicate lower quality. IPS and TPS are decreased with respect to the unconstrained SPNHMM reconstructions, but cMPE reconstructions have considerable less artefacts, which is reflected in higher APS scores. Informal listening tests confirmed the high dependency on correct IMCRA estimates, such that a uniform background noise can be suppressed sufficiently, but more different SE settings are not handled satisfactorily. However, SPNHMM cMPE estimates produce less musical noise than the IMCRA reference.



(a) Raw reconstruction (b) Final estimate $\hat{X}$

Figure 4.5: 1-channel SPNHMM with cMPE (easy SE setting), Model <1> (full example given in Figure B.2).



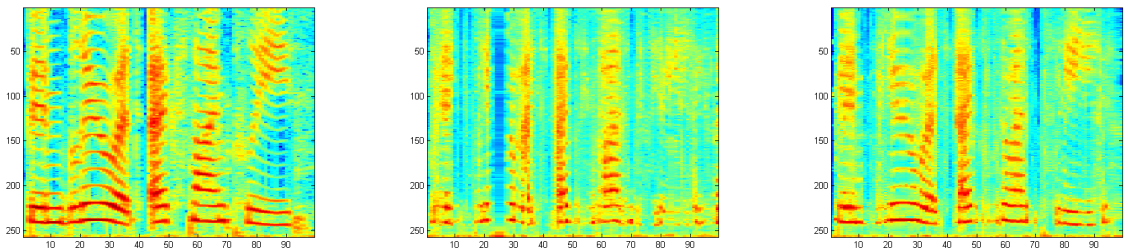
(a) Target speech spectrogram (b) Noisy mixture (c) Final estimate $\hat{X}$

Figure 4.6: 1-channel SPNHMM with cMPE inference (difficult SE setting), Model <1> (full example given in Figure B.3).

4.3 Experiment 2: Indirect mask estimation with 2-channel SPNs

As shown in Experiment 1, it is hard to directly reconstruct the original speech from masks, containing very little information about the clean signal. Two observations led to the second series of experiments: (1) A separate speech prediction $\hat{X}(n,k)$ and a noise prediction $\hat{D}(n,k)$ can be combined by means of a softmask[9] as given in Equation 2.63 (IM-approach). (2) With 1-channel SPNs much information that can be useful for the SPN to reconstruct a clean speech spectrogram is lost because it is marked as noisy by the noise estimator. This is not only an inherent shortcoming of not model-based noise estimators but also a conceptual flaw in the design. With the given models, for the task of SE it seems more appropriate to learn a *relation* between noisy and clean speech.

This can be achieved by letting an SPN learn a joint model of the noisy and clean speech data. Mapping from the mixed to the clean signal requires then running MPE inference on every patch, where the mixed data is given and the clean data is masked. A straight forward approach to this is concatenating the mixture and the clean speech to patches with doubled framewidth but as described in Section 3.4 this does not correctly model the neighbourhood relations between the input variables (compare Figure 4.7(b)). By treating the mixed spectrogram as a noisy representation of the clean speech we can use a 2-channel SPN (described in Section 3.4) which indeed yields better results (compare Figure 4.7(c)).



(a) Target spectrogram (b) 1-channel SPN (c) 2-channel SPN

Figure 4.7: Comparison of clean-speech reconstructions of (a) a 1-channel SPN trained on concatenated patches and (b) a 2-channel SPN trained on layered patches[10] (refer to the text for details).

With 2-channel SPNs we also obtain a new possibility of estimating the noise, namely training a 2-channel SPN with mixed and noise data, so that we can infer the noise given the mixture equivalently as we can infer speech.

To distinguish between the different speech estimations, the direct speech estimates of the network will still be denoted $\hat{X}$, whereas the final estimates of this Experiment are calculated using a softmask (IM-approach) and are denoted $\hat{X}_{\mathrm{SM}}$. Objective results for all models in this experiments are given in Figure 4.8.

[9] For evaluation purposes, we also generated a corresponding binary mask by setting every bin where the corresponding bin in the softmask is greater than 0.5 to 1, and to 0 otherwise.

[10] This finding was part of the preliminary experiments, thus the spectrograms are computed according to Appendix A.

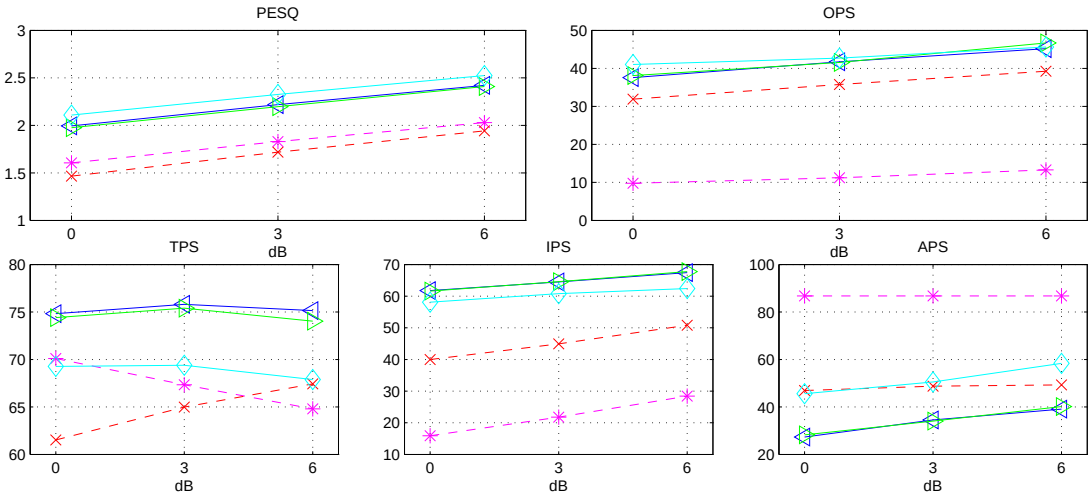


Figure 4.8: Experiment 2: Results overview for 2-channel SPNs, Symbols: (∗) Noisy truth (×) IMCRA reference (◊) 2ch frame-wise SPN (Section 4.3.1) (◁) 2ch SPNHMM (Section 4.3.2) (▷) 2ch SPNHMM with cMPE (Section 4.3.3).

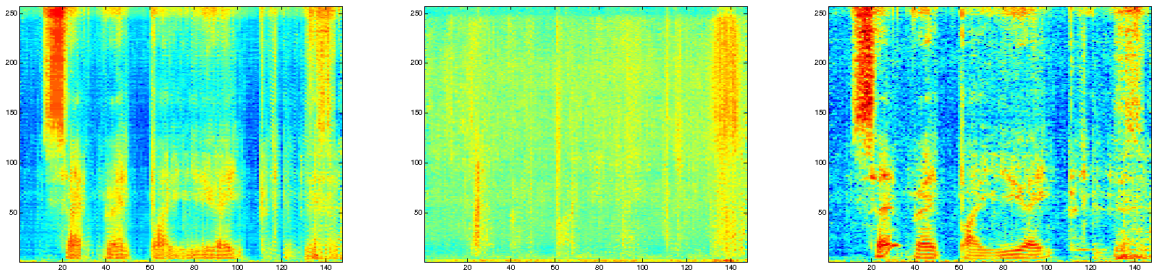
4.3.1 Indirect mask estimation with two 2-channel frame-wise SPNs

Due to excessive training times (see Section 4.5) for the models presented in this section we only used one quarter of the available training data (i.e. 100 utterances) and only report results for the tasks SDc. From the various models trained, results in this section are reported for the two 2-channel SPNs of the following configuration

ID	Model	Type	Size	C	W, H	γ	δ
<2>	2ch frame-wise SPN	M2S	100	-	4,2	12	20
<3>	2ch frame-wise SPN	M2N	100	-	4,2	20	20

Table 4.3: Model parameters of the 2-channel frame-wise SPNs used in Experiment 2.

Under simple SE conditions the raw speech estimates of <2> are visually already similar to the target speech source, with a low amount of noise (Figure 4.9(a)). Combining the speech estimates with the noise estimates of model <3> (Figure 4.9(b)) clearly improves the estimation, although partially wrong noise estimates can also mask correct speech estimates in the final estimate. Moreover, the network mostly fails to reconstruct speech parts that were masked by noise. Figure 4.10 shows the ISM and estimated softmask for the easy SE setting.

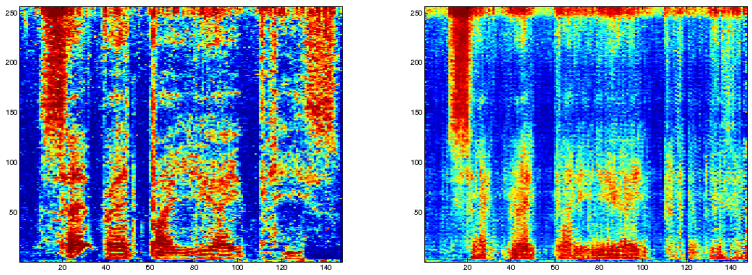


(a) Speech estimate $\hat{X}$ (b) Noise estimate $\hat{D}$ (c) Final estimate $\hat{X}_{SM}$

Figure 4.9: 2-channel frame-wise SPN estimates (easy SE setting), Models: <2><3>.

In more difficult settings it can be noted, that the noise estimates, although very coarse, often

correctly approximate the location of the interfering sources. Speech estimates of <2> also exhibit a very coarse structure, but without any signs of the interfering noise.



(a) ISM (b) Estimated SM

Figure 4.10: Easy example SE setting for a 2-channel frame-wise SPN (full examples given in Figures B.4 and B.5).

The former observations are also reflected in the scores (Figure 4.8): In terms of PESQ, frame-wise SPNs outperform both baselines by a large margin (average relative improvement by half a point) and also show consistently higher OPS values. TPS values indicate that the target is preserved as least as good as for the two baselines but most notably, IPS values show a significant amount of interference suppression where the artefact level is comparable to IMCRA.

Our informal listening tests showed that the combination of the two estimates often does not contain many audible artefacts and effectively attenuates the complex interference, sometimes even in difficult SE environments.

4.3.2 Indirect mask estimation with two 2-channel SPNHMMs

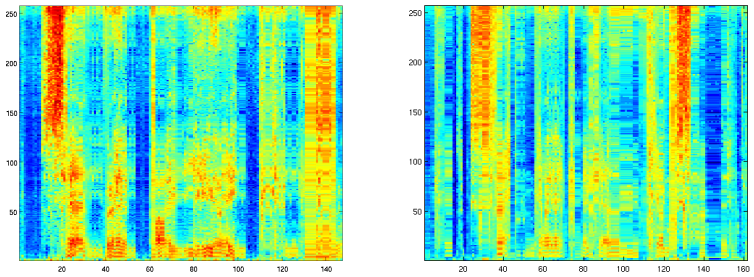
From all the models trained, results in this section are reported for the following models

ID	Model	Type	Size	C	W, H	γ	δ
<4>	2ch SPNHMM	M2S	100	64	1,1	20	20
<5>	2ch SPNHMM	M2N	100	64	1,1	20	20

Table 4.4: Model parameters of the 2-channel SPNHMMs used in Experiment 2.

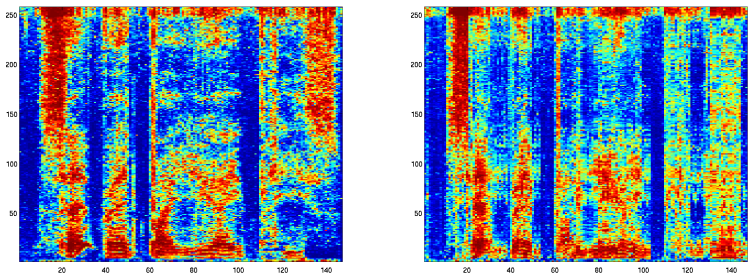
The speech estimates $\hat{X}$ of SPNHMMs already exhibit very speech-like spectrogram structures (e.g. Figure 4.11(a)), although they also become very coarse in more difficult SE settings (e.g. Figure 4.11(b)). The noise estimates of model <5> generally show a bit more structure than the ones of model <3>, and although they also often capture the main spectral locations of interfering sources, noise estimates are very coarse and unsmooth. Figure 4.12 shows the ISM and estimated softmask for the easy SE setting.

Regarding the objective measures, 2-channel SPNHMMs also consistently outperform both baselines in terms of PESQ-scores and have a slightly better performance in OPS-values (Figure 4.8). Suppression of interferences is also good (IPS) but it comes at the cost of introducing quite a lot of artefacts (low APS values). Most notably, 2-channel SPNHMMs outperform all other 2-channel models and baselines by a large margin in TPS-values, indicating that the networks partly learned how to correctly extract speech and noise from the mixture.



(a) Speech estimate $\hat{X}$ (easy) (b) Speech estimate $\hat{X}$ (difficult)

Figure 4.11: 2-channel SPNHMM speech reconstructions of the network for the easy and difficult SE setup, Model: <4>.



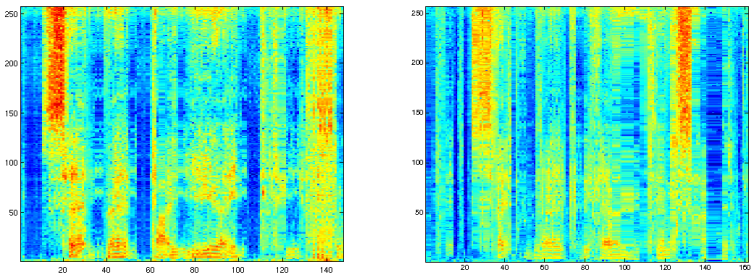
(a) ISM (b) Estimated SM

Figure 4.12: 2-channel SPNHMM softmask comparison (easy SE setting), Models: <4><5> (full examples given in Figures B.6 and B.7).

4.3.3 Indirect mask estimation with two 2-channel SPNHMMs and cMPE

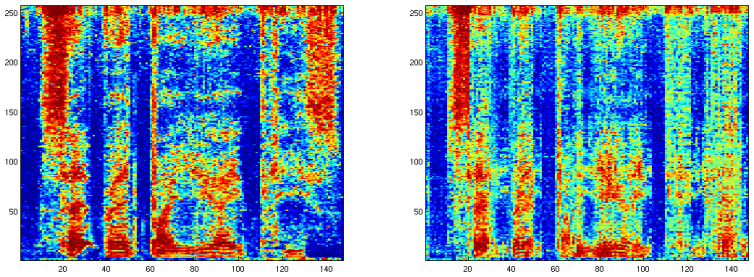
Inspired by the good results of cMPE with 1-channel SPNHMMs, we also used it for 2-channel SPNHMMs. The models used for this task are the same as in Section 4.3.2, i.e. model <4> and model <5>. 2-channel SPNHMMs do not suffer from high-energy reconstruction, thus the application of cMPE is not immediately evident. On the other hand, cMPE imposes a reasonable constraint on the output of the SPN, which might help to correctly identify clean speech structures in some cases.

As shown in Figures 4.13 and 4.12, reconstructions with cMPE tend to be less blocky but do not in general recover more of the masked clean speech structure from the mixture. Overall, the cMPE speech reconstructions are very similar to the ones obtained without cMPE, which can also be seen from the objective scores that are all within the same domain. Figure 4.14 shows the ISM and estimated softmask for the easy SE setting. The only main difference we noted was that the DT-estimates with cMPE (Figure 4.13) achieved better scores than without cMPE. This however is not surprising as the cMPE estimates naturally exhibit more structure in parts of the spectrogram where the SPN is very unsure about the reconstruction.



(a) Speech estimate $\hat{X}$ (easy) (b) Speech estimate $\hat{X}$ (difficult)

Figure 4.13: 2-channel SPNHMM cMPE speech reconstructions of the network for the easy and difficult SE setup, Model: <4>.



(a) ISM (b) Estimated SM

Figure 4.14: 2-channel SPNHMM softmask comparison (easy SE setting) with cMPE, Models: <4><5> (full examples given in Figures B.8 B.9).

4.4 Model comparison

In this section we compare the results of the experiments and put them in a broader context. Furthermore we analyse the impact of other parameters as different masks and reverberant speech data.

Evaluating all results obtained in the different experiments, the first thing to note is that the two baselines are rated rather differently by PESQ and PEASS (Figure 4.15). Whereas in PESQ the IMCRA baseline even looses by a small margin, OPS heavily penalizes the unfiltered signal. Although this big difference might be caused by the fact that the noisy truth is actually an edge case, we PESQ and OPS appear to have quite different underlying concepts of speech quality.

Comparing the results of Experiment 1 (DT-approach) and Experiment 2 (IM-approach) summarized in Figure 4.15, the 2-channel approaches consistently outperform all models that rely on a separate noise estimate (Experiment 1). This however comes at the trade-off that TPS scores are smaller, which is reasonable as the models from Experiment 1 built their estimates on top of a set of TF-bins, that already has a high likelihood of being speech. In terms of APS, 1-channel SPNHMMs with cMPE and 2-channel frame-wise SPNs are comparable to IMCRA, which we established in informal listening tests to be an acceptable amount of artefacts. Observing all results, there seems to be a trade-off between TPS and APS, which is consistent with our visual observation of the reconstructed spectrograms. Overall, the results suggest that the frame-wise SPN yields the best overall quality. Our informal listening tests confirmed this result and it appears that a major factor for a pleasant listening experience is the ratio of TPS/APS.

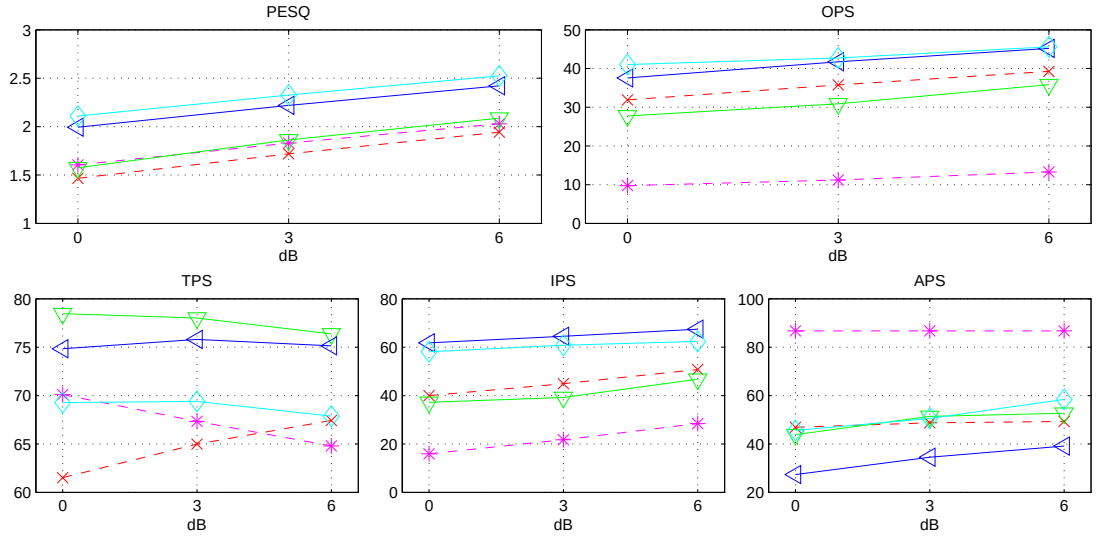


Figure 4.15: Comparison of Experiment 1 and Experiment 2, Task: SDc, Symbols: (∗) Noisy truth (×) IMCRA reference (▽) SPNHMM with cMPE (Section 4.2.2) (◇) 2ch frame-wise SPN (Section 4.3.1) (◁) 2ch SPNHMM (Section 4.3.2).

4.4.1 Comparison with other deep models

Results on the task SDr reported in [10] (DM) and [6] (IM) show a clear dominance of IM-GSNs over all other models examined in this work in terms of PESQ scores. Regarding OPS, it can be seen that the DM-GSN approach performs worse than IMCRA, but IM-GSNs still yield the best performance. Comparison with Gauss-Bernoulli RBMs (GBRBMs), constraint GBRBMs (CGBRBMs), HCAEs and MLPs [6] shows that in terms of PESQ scores, SPNs are in line with GBRBMs and loose against CGBRGM and HCAE by a small margin. In term of OPS, MLPs

still perform slightly better but SPNs outperform GBRMs, CGBRBMs and HCAEs.

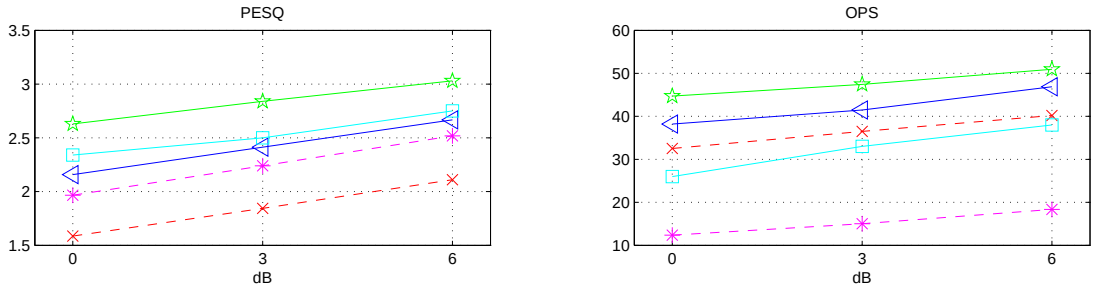


Figure 4.16: Comparison with GSN on Task SDr, Symbols: (∗) Noisy truth (×) IMCRA reference (▷) 2ch SPNHMM (Section 4.3.2) (□) GSN with DM-approach [10] (⋆) GSN with IM-approach [6].

4.4.2 Comparison of direct target and indirect mask estimation approaches

In Experiment 2 we used softmasks for all estimations. Nevertheless we also evaluated the performance of binary masks (IM) and of direct speech estimation (DT). As it can be seen in Figure 4.17, softmasks appear to be superior in PESQ, OPS, TPS and APS and absolute IPS scores are also high. However, this does not mean that BMs are not suitable for the task of SE, but indicates that it is not advantageous to use BMs, having two separate continuous estimates at hand. BMs reduce the problem of SE from an estimation to a classification problem and algorithms with a suitable objective might take precise advantage of that.

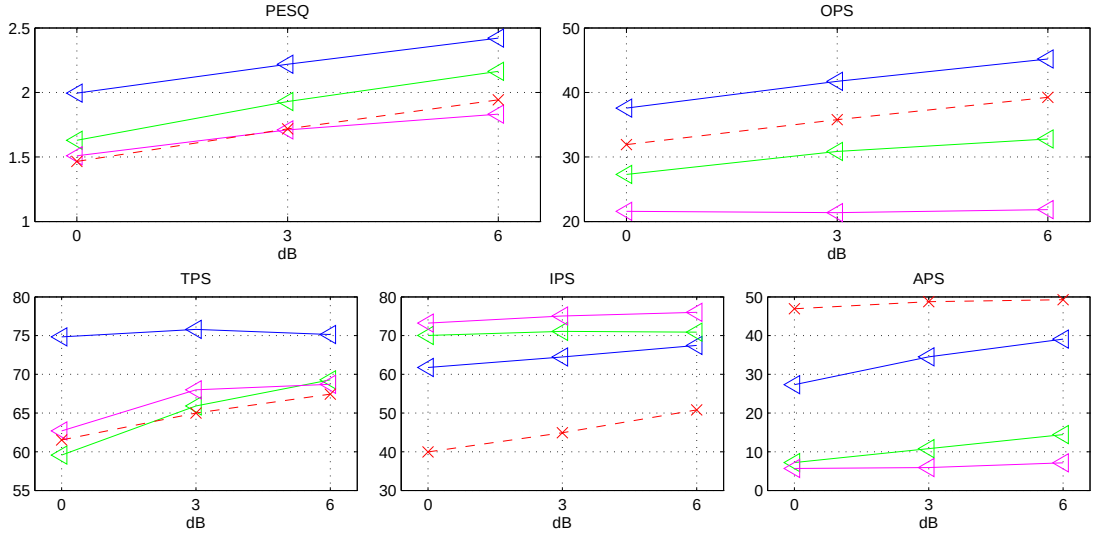


Figure 4.17: Comparison of SM, BM and DT-approach, Symbols: (×) IMCRA reference (◁) 2ch SPNHMM (Section 4.3.2) (blue: SM, green: BM, magenta: TD).

In Section 2.4 we showed that an IM-approach only makes sense with two separate estimates of speech and noise. An intuition to why softmasks work well in comparison to direct speech estimates can be given by assuming a uniform noise estimate $D(n,k) = 1$ which yields $H = \hat{X}/(\hat{X}+1)$. Under this assumption, the speech estimate $\hat{X}$ only acts as a conservative *weighting*[11]

[11] A DT-approach thought as a softmasks under the constraint $\hat{X} \leq Y$ is a linear mapping from $\hat{X}$ to $\hat{X}_{SM}$ that is capped at Y. An IM-approach with a low-energy constant noise estimate predicts in comparison with the linear mapping lower $\hat{X}_{SM}$ for higher $\hat{X}$ and higher $\hat{X}_{SM}$ for lower $\hat{X}$.

of the mixed signal which helps to preserve its natural structure. Therefore using softmasks can be seen as being less error-prone than DT but comes at the expense that the speech and noise estimates have to be in tight correspondence to achieve high attenuation.

4.4.3 Comparison of tasks

Performing SE with speaker-independent model (Figure 4.18, top) decreases the achieved performance on both measures by approximately 15% (although it appears that the SI-dataset is slightly more difficult as IMCRA scores are also decreased). SE on reverberant data on the other hand seems easier as PESQ scores for SPNHMM and IMCRA are slightly increased (Figure 4.18, bottom). OPS scores are largely the same.

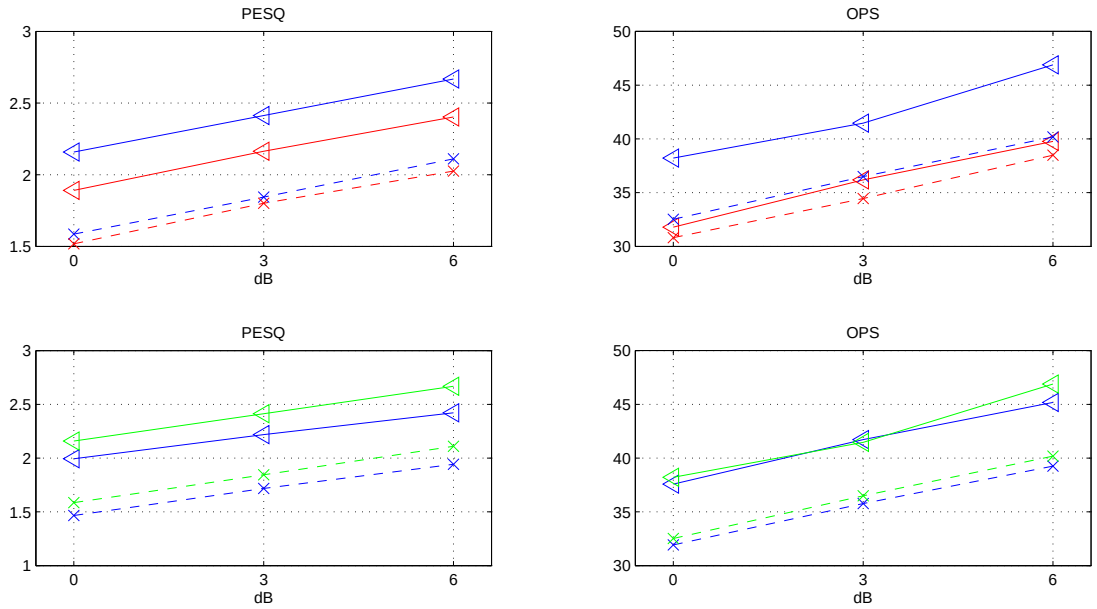


Figure 4.18: Comparison between tasks, (Top) SDc vs. SIc (Bottom) SDc vs SDr, Blue: SDc, Red: SIc, Green: SDr, (×) IMCRA reference (◁) 2ch SPNHMM, Section 4.3.2.

4.5 Discussion of SPNs

In the previous sections we showed that SPNs can provide reasonable ways for tackling the problem of speech enhancement. Nevertheless, all tested models showed specific advantages and drawbacks, which are the focus of this section.

4.5.1 1-channel SPNs

Although 1-channel SPNs are trained on clean speech containing many low-energy TF-bins, when given a mask with high-energy TF-bins only, they do not correctly reconstruct the low-energy parts of the clean speech[12] (see Figure 4.4(c)). Further investigation revealed that the distribution of low-level features, i.e. the fist layer of product-nodes joining 2 Gaussian probability nodes of different scope, has a high peak for features that model the interaction of similar

[12] This behaviour could also be observed for 1-channel frame-wise SPNs, although they were not formally evaluated during this work.

magnitudes. As it can be seen in Figure 4.19, there are hardly any features that model the interaction of two Gaussian nodes with a difference of means of 2 or more, which roughly corresponds to the difference of high- and low-energy TF-bins on normalized log-spectrogram data. Thus, given a mask with only high-energy TF-bins, the majority of the low-level features will vote for a reconstruction with approximately equal energy. This feature distribution however naturally emerges from the input data as the first layer of product-nodes always joins neighbouring TF-bins, which in our setup (compare Section 4.1.1) are highly correlated [48].

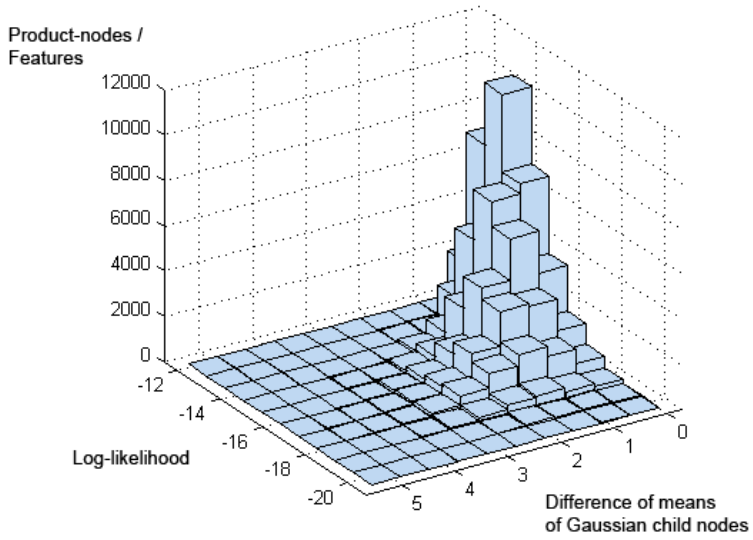


Figure 4.19: Distribution of the features modelled by product nodes on the third layer. Each product-node has two Gaussian distribution child nodes. The log-likelihood of a product node is a priori given by the sum of the log-weights of all paths going from the root node to the product-node.

4.5.2 2-channel SPNs

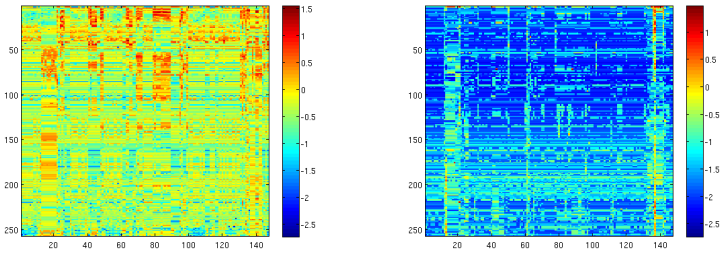
Although the number of Gaussian distribution nodes increases quadratically with the number of input variables, even alotting 20 Gaussian nodes per TF-bin and layer (yielding 400 Gaussian nodes per TF-bin for both layers) did no seem to pose a problem for SPN training and reconstruction. Especially with 2-channel SPNHMMs, fine-grained reconstructions could be achieved.

One reason for the inferior performance of 2-channel SPNs in comparison with GSNs is presumptively their different objective. GSNs as used in [6] are trained to reconstruct a clean speech signal from a speech representation that is corrupted by noise, where the noise happens to be the one that is already present in the noisy mixture. It is not clear to what extend GSNs learn the *conditional* distribution $\mathrm{P}(X \mid Y)$ or the *marginal* $\mathrm{P}(X)$, but it appears to be a useful representation for solving the mapping problem between the two representations $Y \rightarrow X$. 2-channel SPNs on the other hand model the *joint* distribution of the mixture and speech $\mathrm{P}(X, Y)$, which is then used to infer the clean speech spectrogram that provides the most probable explanation (MPE) under the joint given only the mixture. In the context of the distinction given in Section 2.4, one can see our 2-channel approach as supervised and generative. It has widely been shown [49] that models with a discriminative objective generally perform better on discriminative tasks than generative models and according to the deep-learning paradigm [26, 45], generative modelling can provide vital cues for solving discriminative tasks, but does not yield optimal performance by itself. Although our approach opens the door for methods similar to

[26, 45], we do not further exploit this property, which we assume is the main reason for the difference in performance.

2-channel frame-wise SPNs

Frame-wise SPNs as used in this work are a purely heuristic approach. Regardless of their respectable performance on the tasks, they show some severe problems: The first is that although constraint optimization does yield good results with SPNHMMs, despite best efforts, it showed not to provide reasonable results for frame-wise SPNs. Further research showed, that the problem is somehow related to input data selectivity: Two small models with a framewidth of 1 were trained with the same amount of data, where one model was given only similar patches (corresponding to one state of an HMM), whereas the other was trained with the full variety of input data. Figure 4.20 shows that the former model provides reasonable results in terms of log-magnitude, whereas the latter returns a very low-energy reconstruction.



(a) Selective data (b) Unselective data

Figure 4.20: Comparison of cMPE reconstructions of two SPN models trained with (a) a limited or (b) the entire dataset.

The second problem is that although the 2-channel frame-wise SPNs were trained with a sufficient amount of Gaussians (i.e. 20) to model the mixture and the clean speech, the resulting reconstructions are very coarse and uniform over large parts of the spectrogram.

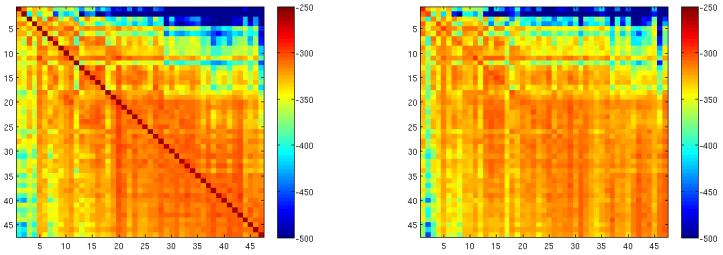
The third problem is that training 2-channel frame-wise SPN is considerably more time-consuming then training 1-channel frame-wise SPNs or SPNHMMs. This has a couple of reason: First, in a 2-channel SPN the number of Gaussian probability nodes is quadratic in the number of input variables. Second, we only trained models with $W > 1$, thus the number of input variables is also increased leading to a polynomial increase of the network size. Third, in contrast to SPNHMMs, one model is trained on all available patches but the training of one SPN cannot be trivially parallelized.

2-channel SPNHMMs

Data selectivity is a quantity that expresses the narrowness of a selection with respect to the entire data-set.

The underlying assumption of an SPNHMM is that the subSPNs show sufficient data selectivity towards the data they were trained on. From a more generative perspective this is: At inference stage the subSPN has to recognize to what extend it was responsible for generating the presented mixture, although the mixture is a combination of an unseen clean speech and an unseen noise component. This selectivity is important because the Backward-Forward algorithm relies on the likelihood differences between the models to calculate a weighted trajectory (middle plots in Figure 4.22). If the subSPNs are led astray too much by the mixture, the 'wrong' subSPNs will be chosen to reconstruct the clean speech.

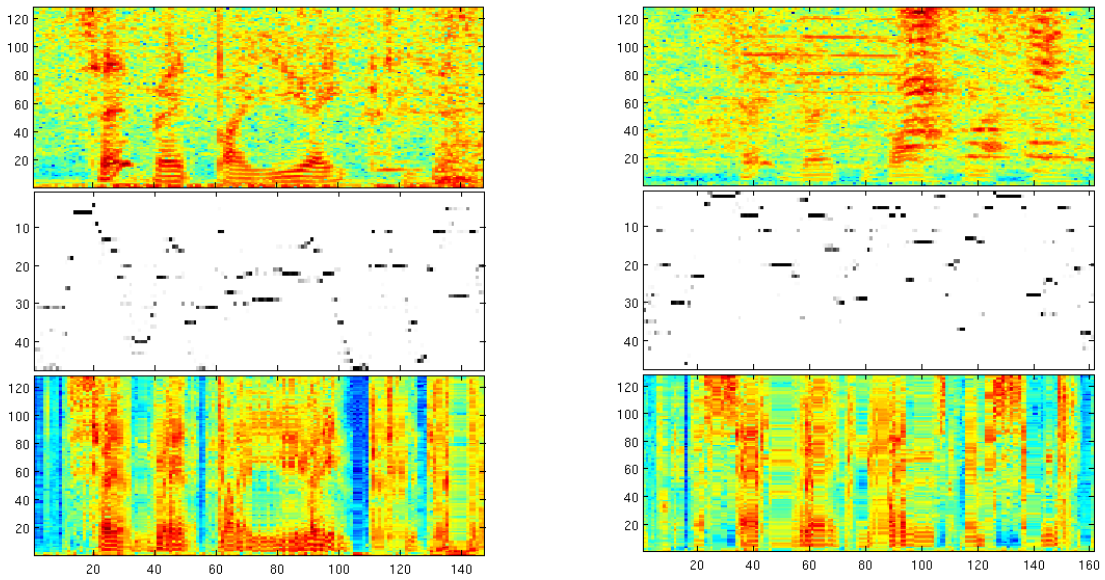
To investigate how subSPNs perform on this difficult sub-task, one can first observe their selectivity towards the training-data, i.e. each subSPN should give high likelihoods to the mixture-samples it was trained on, and lower likelihoods to all others. As shown in Figure 4.21(a), this is the case. If we now perform the same test with mixture-samples where we replaced the known noise with an unseen noise, we see that data-selectivity heavily decreases and is only given to some extend for models trained on fricatives and strong voiced phonemes (corresponding to the top rows in Figure 4.21(b)). Recognizing a pattern in a mixture of two unseen components is even a harder problem and data-selectivity will likely further decrease. Nevertheless, it appears that in combination with the Forward-Backward algorithm, the calculated weighted trajectories in Figure 4.22 (middle plots) are quite reasonable and that the produced acoustic artefacts in difficult SE-settings are rather caused by insufficient MPE reconstructions than insufficient data-selectivity.



(a) Mixture (b) Mixture with unseen noise

Figure 4.21: Data selectivity of a 2-channel SPNHMM towards (a) mixtures of the training-set (b) a newly combined mixture with the speech from the training-set but unseen noise, high likelihoods on the main diagonal mean high model-specific data-selectivity; Model: <4>, only 47 of 64 clusters shown[13].

[13] The other clusters correspond to silent frames and almost never give high likelihoods.



(a) Easy SE setting

(b) Difficult SE setting

Figure 4.22: Weighted trajectories of a 2-channel SPNHMM in the easy and difficult SE setting, (Top) Mixture (Middle) Weighted trajectories (weighted responsibilities of every subSPN for every sample) (Bottom) MPE reconstruction; Model: <4>, only 47 of 64 clusters[13] and 128 of 256 frequency-bins shown.

4.5.3 Overall observations

SPNs are, independent of the concrete model they are used in, quite unsensitive to variation in parameters. This means on the other hand that many of the models that we trained with different parameters yielded similar test results. No clear indications were found during this work, which of the known parameters could potentially significantly increase their performance.

Another observation that was not further pursued was that PD SPNs did not consistently react on variations of the sparse prior. Although an l_0-prior enforces model sparsity, the PD architecture tends to build large and very deep models[14], what raises the question of their ability to generalize well the data. One indicator for why these models do not severely overfit despite their size is that many sum-nodes only have one child (with a weight of 1), thus they only propagate information from lower to higher layers[15].

[14] The models used in this work had an average depth of 262 layers.

[15] Of the models observed, more than 90% of the sum-nodes in SPNHMM subSPNs and about 70% of the sum-nodes in frame-wise SPNs only had one descendent. This fact could potentially be used for compressing the networks by attaching all independent distributions of smaller scope directly to a product node of larger scope.

5 Conclusion

Sum-Product Networks (SPNs) are a new type of deep model, that can represent complex probability distributions by combining simple distributions via sum- and product-nodes. They can be trained to learn a statistical model of some given data, such as clean speech.
Speech enhancement (SE) is the task of separating and reconstructing speech from a mixture signal that has been corrupted by additive noise, such as car-noise or restaurant-noise. This has practical relevance e.g. in telecommunication as SE can improve the quality and intelligibility of speech.
In this work we applied SPNs to the problem of SE within two experiments: In the first experiment we trained an SPN on clean speech data and used a separate noise estimator to provide a speech probability mask. Regions of low speech probability are then reconstructed by the SPN (DT-approach, see Section 2.1.1). In the second experiment we trained one SPN to represent the joint probability distribution of the noisy mixture and the clean speech. Another SPN was trained on the noisy mixture and the additive noise. To account for the new structure of the input data we developed a modification of an SPN which we called 2-channel SPN. For reconstructing the clean speech we let both models infer an estimate of the clean speech and the noise respectively. These two estimates are then combined by means of a softmask which can subsequently be used to filter the clean speech from the mixture. This method we denote as IM-approach (see Section 2.1.1). For comparison of our results we used the original mixture and a de-noised version of the signal provided by IMCRA. Furthermore, we also compared our results to other models, most importantly to general stochastic networks (GSNs).
The main conclusions from the experiments are the following:

- *Model performance*: Although SE is a difficult task, SPNs manage to provide reasonable results, especially the IM-approach with 2-channel SPNs. Objective scores indicate increased quality compared to the noisy mixture and a non model-based estimator (IMCRA). Nevertheless even the best SPNs are outperformed by GSNs by a large margin.

- *Applicability of generative modelling*: SPNs generatively model the distribution of its inputs, which makes them especially suitable for tasks that require a joint probability density function. Due to their generality, they can also be used for solving discriminative tasks, such as mapping a mixture to clean speech. Nevertheless, they will naturally be outperformed by models with a more constraint objective, such as GSNs.

- *SE approaches*: We showed that the IM-approaches consistently outperform the DT-approaches and that softmasks work well for combining two different estimates.

5.1 Outlook

SPNs are becoming increasingly popular in the area of deep-learning. Some interesting research directions for future work are:

- *Other objectives*: As shown in [44], SPNs can also be trained discriminatively. This approach could be generalized to make SPN available for other objectives such as needed for e.g. discriminative-finetuning.
- *Regularization*: As described in Section 4.5.3, the structure of PD SPNs is quite cumbersome. One approach taken in [50] and [43] is to learn the structure of a network by the data but another important point seems to be regularization, where methods as dropout [32] or pruning could be adapted for SPNs.
- *Domain knowledge*: Including domain knowledge might help the network to extract better problem-specific features. For SE, this could be done by one of the following ways: (1) Preprocessing of the data, e.g. with a gammatone filterbank (2) Incorporating other objectives that explicitly optimize speech-specific measures or (3) Encoding of basic knowledge about speech productions into the structure of an SPN.
- *Generative problems*: Sampling from SPNs is very easy and during this work preliminary experiments indicated that the internal representations of SPNs are very accurate. This could potentially be used for tasks as e.g. speech synthesis.
- *Model combinations*: SPNs could form a building block of more advanced models such as FHMMs (see Section 2.4.1).

A Further research topics

Much research performed for this work was devoted to preliminary experiments. This appendix gives a brief overview over two selected methods.

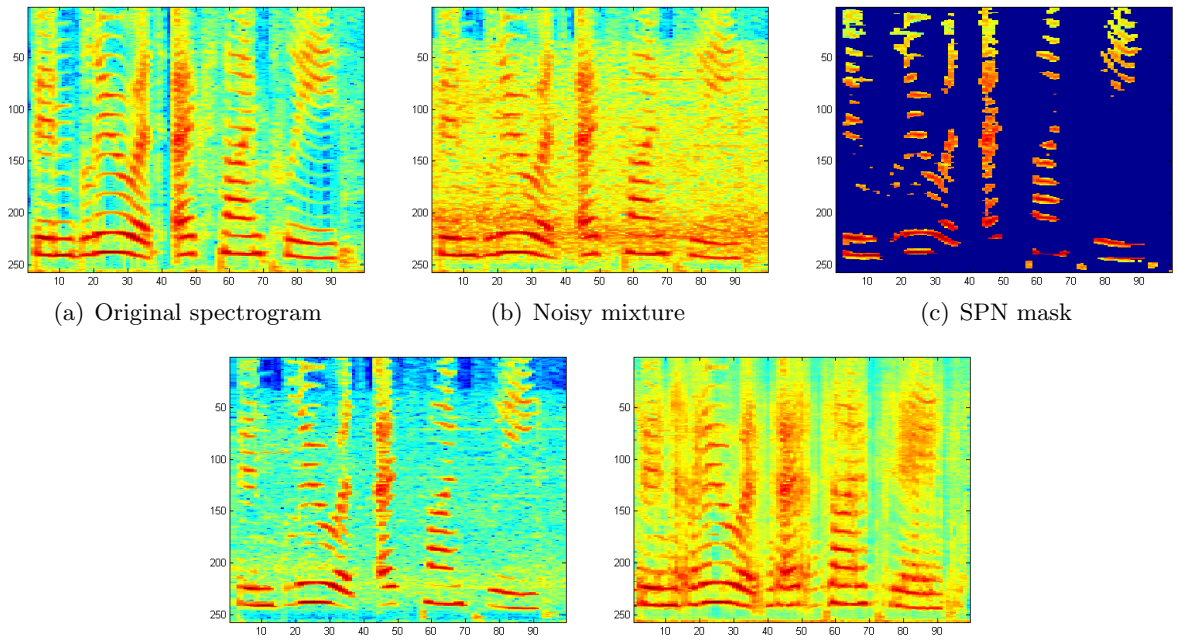
All experiments described in this section were performed on clean speech data of the GRID [51] corpus where the input data was downsampled to 8kHz, before computing a 512 point Fourier Transform.

A.1 Low Energy Bias (LEB)

1-channel SPNHMMs tend to reconstruct high energy magnitudes, given the TF-bins with high probability of being speech[16] (compare 4.5.1). A first try to remedy this problem was to introduce a low energy bias on the MPE reconstructions, which follows the same underlying principle as the constraint MPE: Gaussian nodes of a marginalized variables return a likelihood of 1, as this corresponds to summing over all possible values of this distribution (see 3.4). If a Gaussian node is given a value higher than 1, features and mixtures of features containing this node will also receive higher likelihoods. For biasing specific variables we introduced coefficients which define the amount of LEB for each pixel, i.e. the likelihood boost of marginalized Gaussian nodes with lower means.

Two main approaches have been tried: (1) Introducing a general low-energy bias on all non-evidence TF-bins (Figure A.2) and (2) Placing a low-energy bias on TF-bins around the evidence bins (Figure A.3). Biasing the reconstructions towards lower values yielded lower-energy results, although there were several problems with this approach: (1) The effect of LEB is data-dependent and the coefficients are very sensitive to tune. It can be seen in Figure A.3(b) that despite of giving TF-bins surrounding the speech evidence a low bias, the bias primarily acts on the whole patch. (2) Using a low-energy bias is a purely heuristical counter-measure against the fundamental problem of how PD SPNs model speech probabilities. (3) Performance results showed a consistently lower OPS than the reference IMCRA-processed signal at a computational cost increased by the factor 10-100.

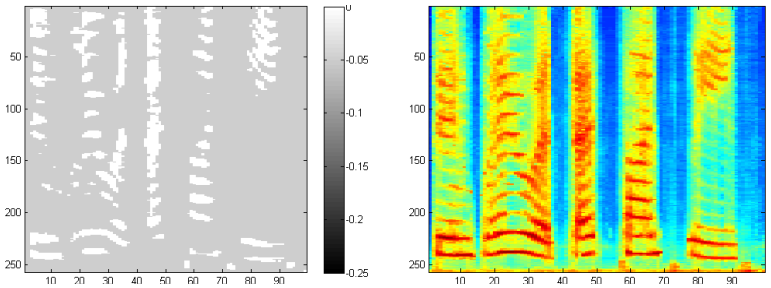
[16] As confirmed on examples of this dataset, this is also true for 1-channel frame-wise SPNs.



(a) Original spectrogram (b) Noisy mixture (c) SPN mask

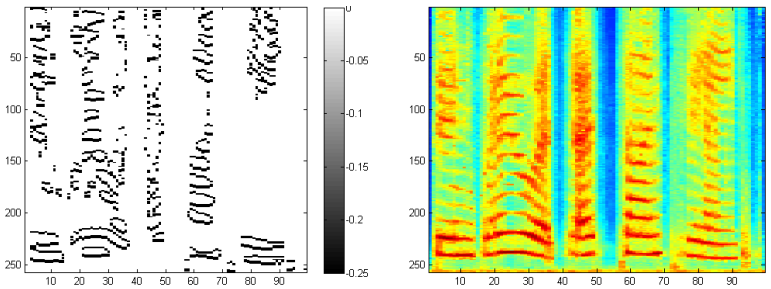
(d) IMCRA reference (e) SPN reconstruction without LEB

Figure A.1: 1-channel SPN reconstruction with low-energy bias (LEB).



(a) LEB input (b) SPN reconstruction

Figure A.2: 1-channel SPN reconstruction with uniformly distributed LEB coefficients.



(a) LEB input (b) SPN reconstruction

Figure A.3: 1-channel SPN reconstructions with speech-surrounding LEB coefficients.

A.2 Gradient ascent

Inference on a mixture sample in an 1-channel frame-wise SPN trained on clean speech data will yield a low likelihood, but we can use the likelihood gradient to hillclimb to a state of higher likelihood. Ideally, this state will correspond to a less noisy version of the noisy input.

The likelihood gradient in an SPN can be easily calculated and is given by

$$\begin{aligned}\frac{\partial S(\boldsymbol{x})}{\partial X_i} &= \sum_{k\in\mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})}\frac{\partial S_k(\boldsymbol{x})}{\partial X_i} \\ &= \sum_{k\in\mathrm{pa}(i)} \frac{\partial S(\boldsymbol{x})}{\partial S_k(\boldsymbol{x})}\frac{\partial \mathcal{N}(X_i \,|\, \mu_i^k, \sigma_i^k)}{\partial X_i}\end{aligned} \tag{A.1}$$

where in the log-domain the gradient of a Gaussian becomes a linear function

$$\frac{\partial \log\big(\mathcal{N}(X_i \,|\, \mu_i^k, \sigma_i^k)\big)}{\partial X_i} = (x_i - \mu_i^k)/(\sigma_i^k)^2 \tag{A.2}$$

For optimizing the likelihood we used the L-BFGS (Limited-memory Broyden-Fletcher-Goldfarb-Shannon) algorithm. As it can be seen in Figure A.4, the optimization always converged to a very similar parameter-space corresponding to silence. Although this provides in insight into to structure of the likelihood surface, this approach is conceptually flawed as it needs to find good local maximum near the starting point, that must not be global maximum (which is the same for every patch).

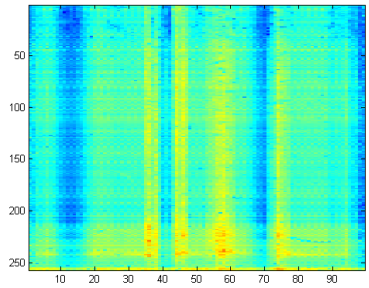


Figure A.4: Results of running gradient ascent method on the noisy input spectrogram given by Fig. A.1(b).

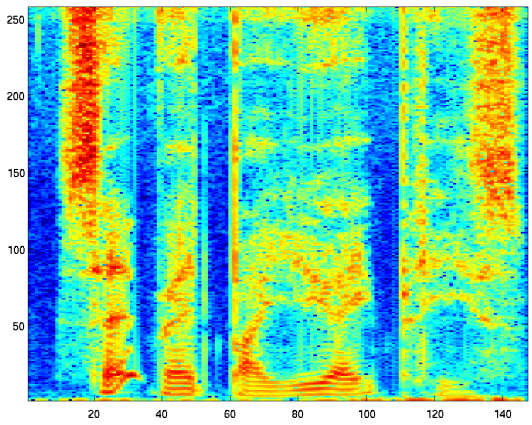
B Plots

In this appendix, full example plots of all models described in Chapter 4 are given. Table B.1 gives an overview over all models

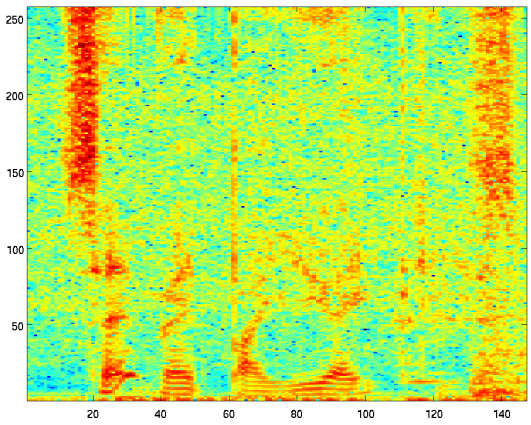
ID	**Model**	**Type**	**Size**	$\boldsymbol{C}$	$\boldsymbol{W}$, $\boldsymbol{H}$	$\boldsymbol{\gamma}$	$\boldsymbol{\delta}$
<1>	1ch SPNHMM	S	400	64	1,1	20	20
<2>	2ch frame-wise SPN	M2S	100	-	4,2	12	20
<3>	2ch frame-wise SPN	M2N	100	-	4,2	20	20
<4>	2ch SPNHMM	M2S	100	64	1,1	20	20
<5>	2ch SPNHMM	M2N	100	64	1,1	20	20

Table B.1: Overview over all models.

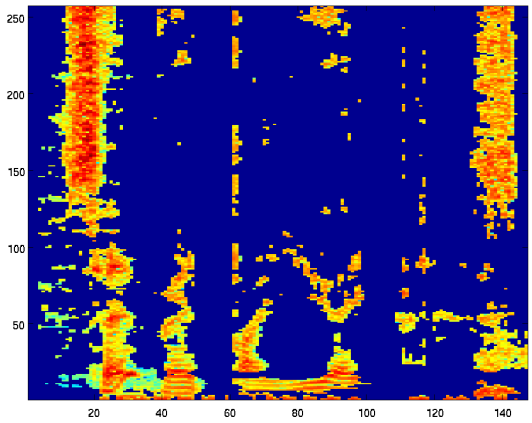
B.1 Detailed figures of Experiment 1



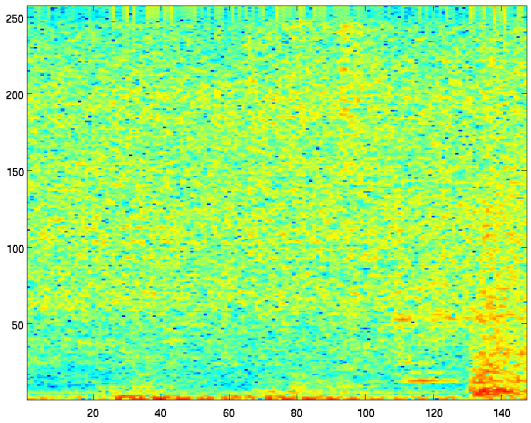
(a) Speech spectrogram



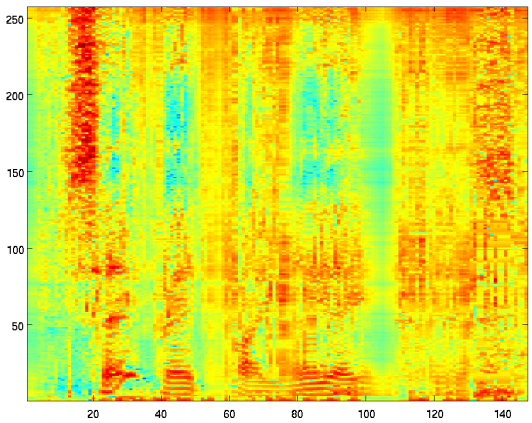
(b) Noisy mixture



(c) SPN mask, blue corresponds to missing bins

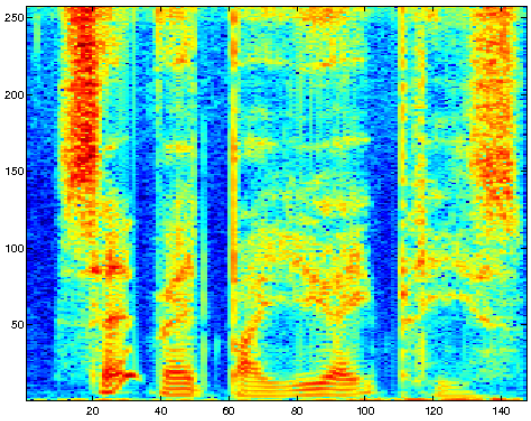


(d) Noise spectrogram

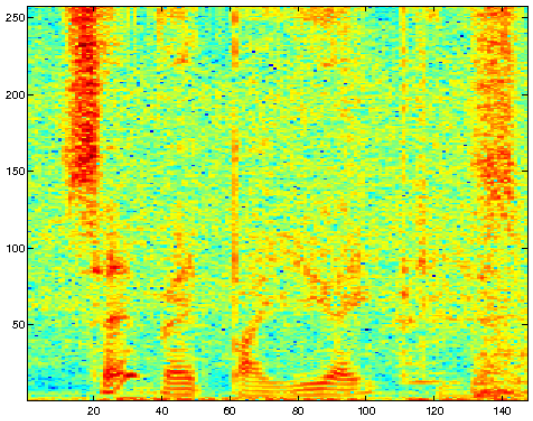


(e) Raw SPNHMM reconstruction

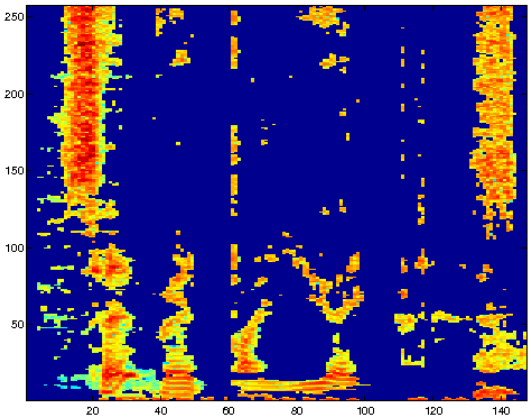
Figure B.1: Experiment 1: Figure showing an easy example SE setting for a 1-channel SPNHMM with unconstrained reconstruction, File: srbu8p, Task: SDc, noise at +0dB, Model: <1>.



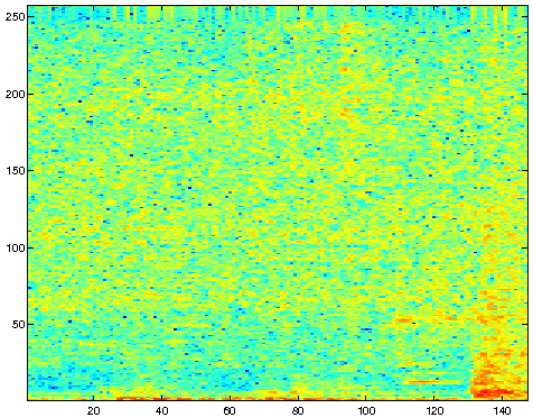
(a) Speech spectrogram



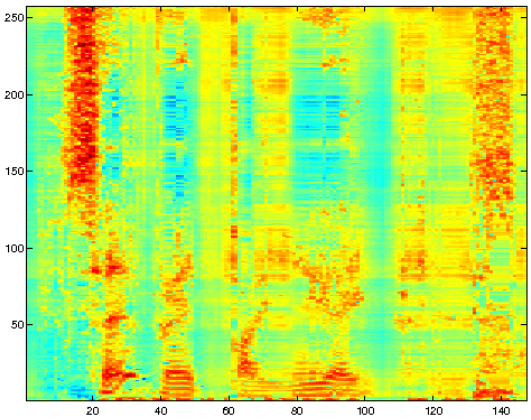
(b) Noisy mixture



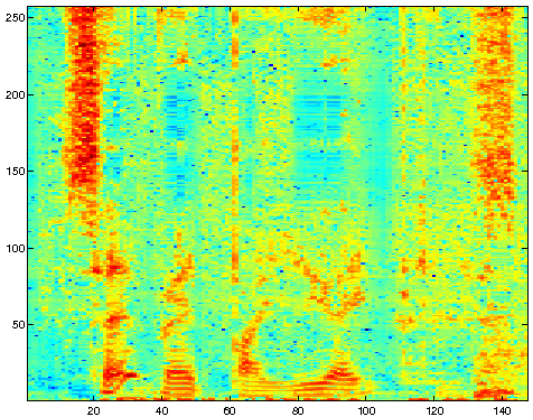
(c) SPN mask, blue corresponds to missing bins



(d) Noise spectrogram

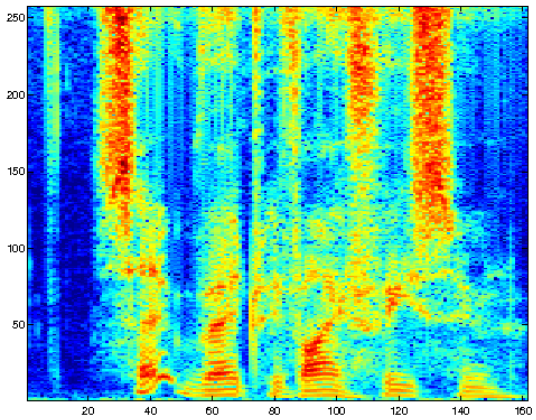


(e) Raw SPNHMM reconstruction

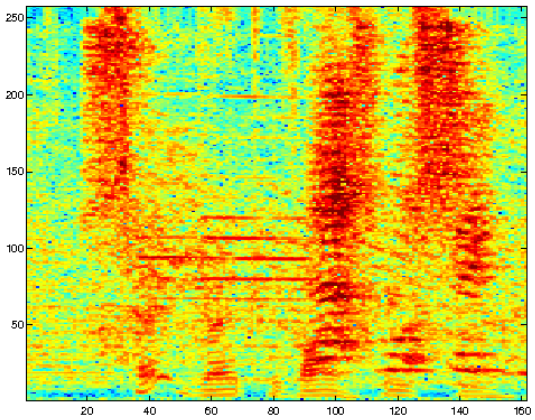


(f) SPNHMM reconstruction, $\hat{X} \leq Y$ enforced

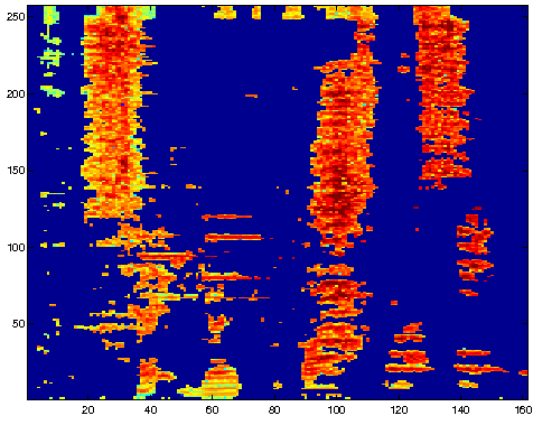
Figure B.2: Experiment 1: Figure showing an easy example SE setting for an 1-channel SPNHMM with cMPE, File: srbu8p, Task: SDc, noise at +0dB, Model: <1>.



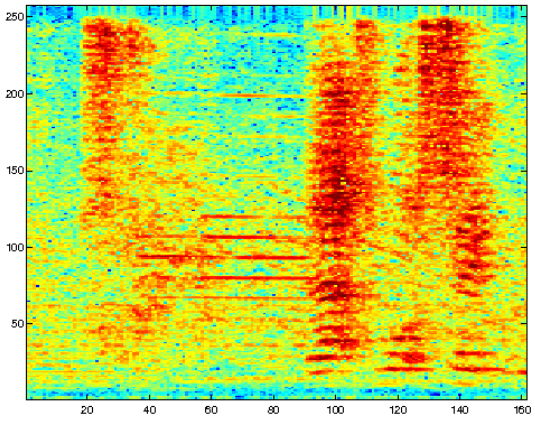
(a) Speech spectrogram



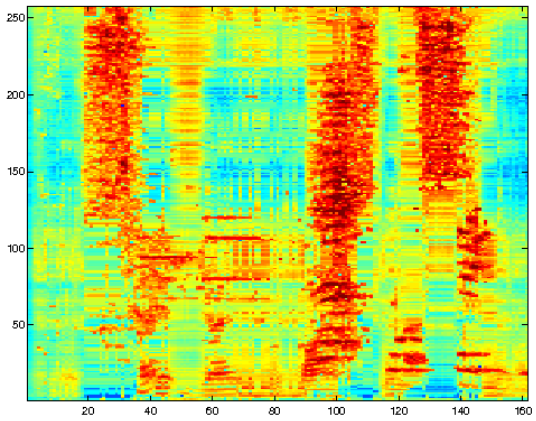
(b) Noisy mixture



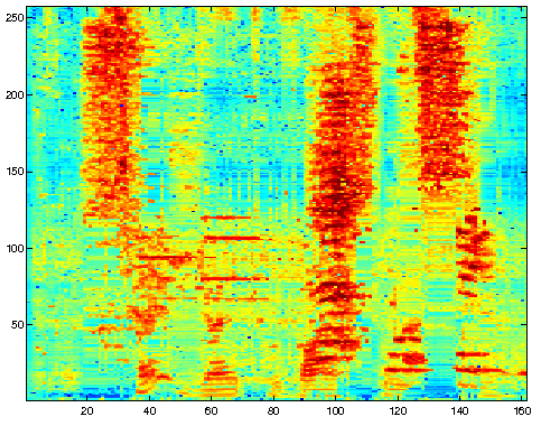
(c) SPN mask, blue corresponds to missing bins



(d) Noise spectrogram



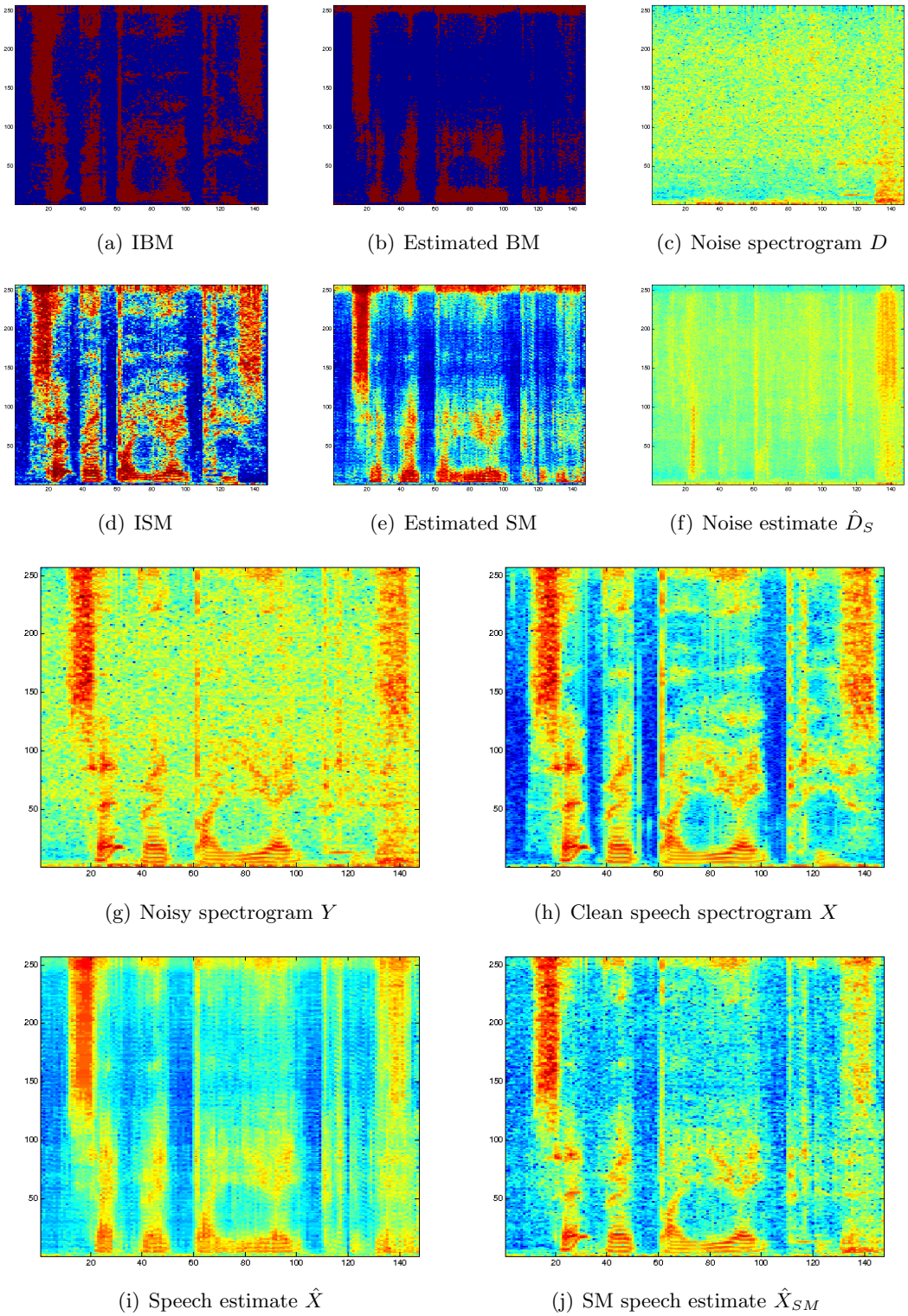
(e) Raw SPNHMM reconstruction with cMPE



(f) cMPE SPNHMM reconstruction, $\hat{X} \leq Y$ enforced

Figure B.3: Experiment 1: Figure showing a difficult example SE setting for 1-channel SPNHMM with cMPE, File: srwi3s, Task: SDc, noise at +0dB, Model: <1>.

B.2 Detailed figures of Experiment 2



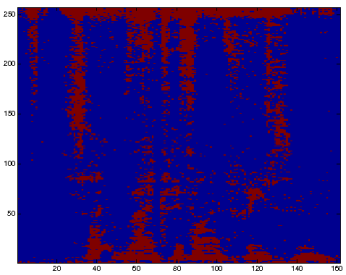
(a) IBM (b) Estimated BM (c) Noise spectrogram D

(d) ISM (e) Estimated SM (f) Noise estimate $\hat{D}_S$

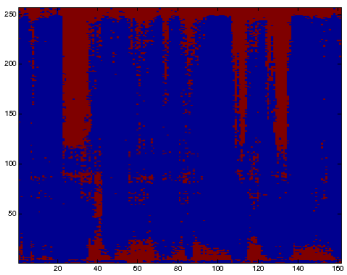
(g) Noisy spectrogram Y (h) Clean speech spectrogram X

(i) Speech estimate $\hat{X}$ (j) SM speech estimate $\hat{X}_{SM}$

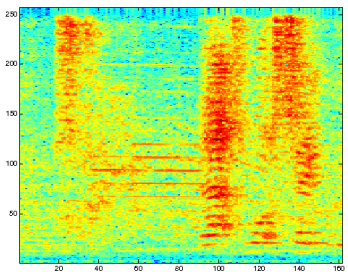
Figure B.4: Experiment 2: Figure showing an easy example SE setting for a 2-channel frame-wise SPN, File: srbu8p, Task: SDc, noise at +0dB, Models: <2><3>.



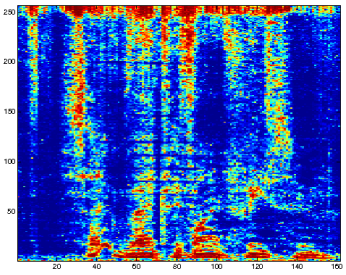
(a) IBM



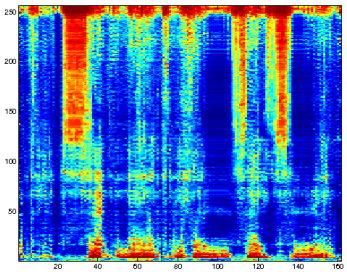
(b) Estimated BM



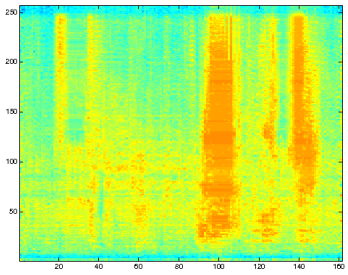
(c) Noise spectrogram D



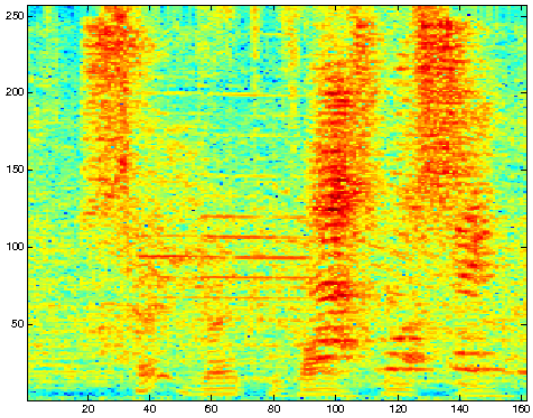
(d) ISM



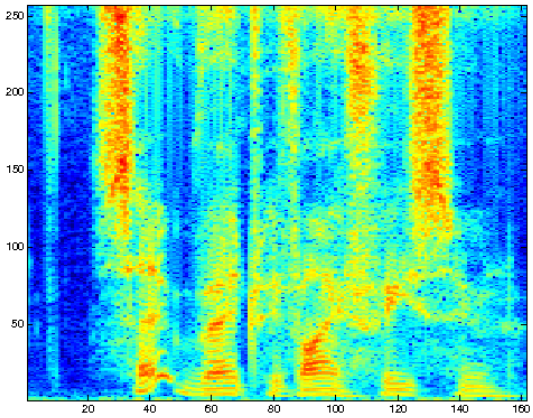
(e) Estimated SM



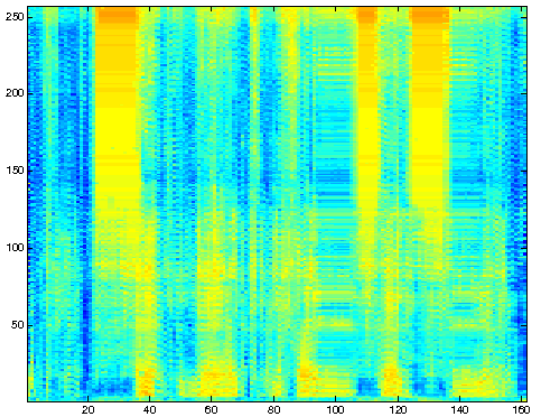
(f) Noise estimate $\hat{D}_S$



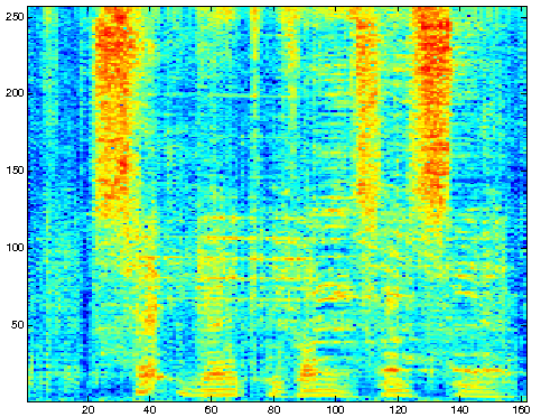
(g) Noisy spectrogram Y



(h) Clean speech spectrogram X

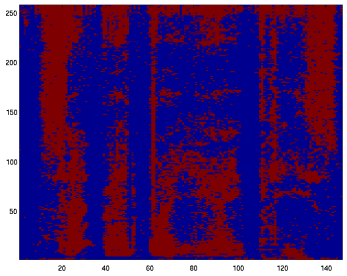


(i) Speech estimate $\hat{X}$

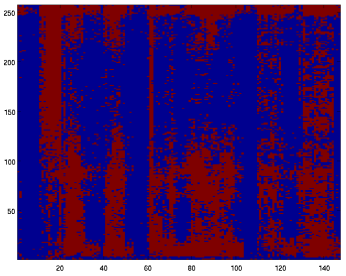


(j) SM speech estimate $\hat{X}_{SM}$

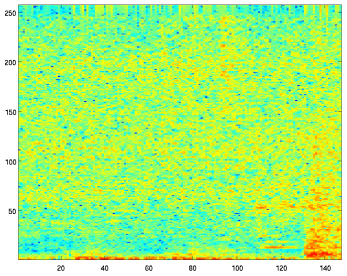
Figure B.5: Experiment 2: Figure showing a difficult example SE setting for a 2-channel frame-wise SPN, File: srwi3s, Task: SDc, noise at +0dB, Models: <2><3>.



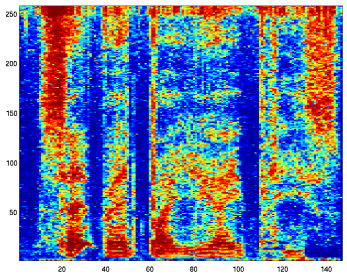
(a) IBM



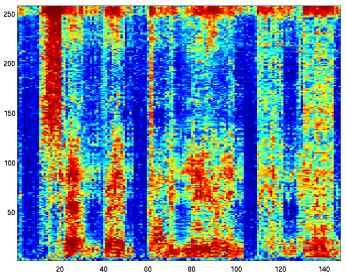
(b) Estimated BM



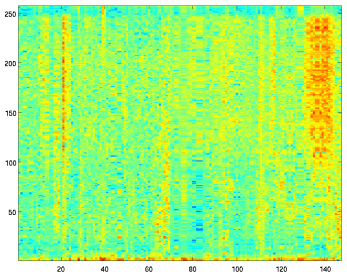
(c) Noise spectrogram D



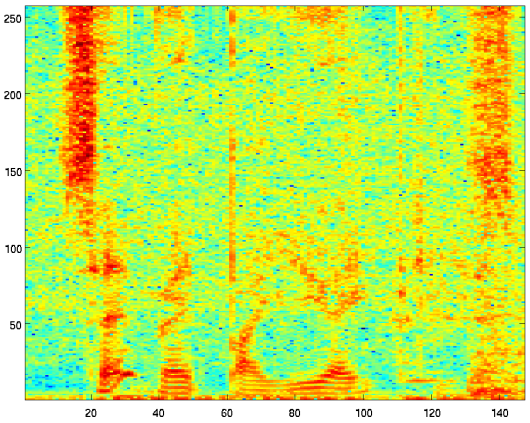
(d) ISM



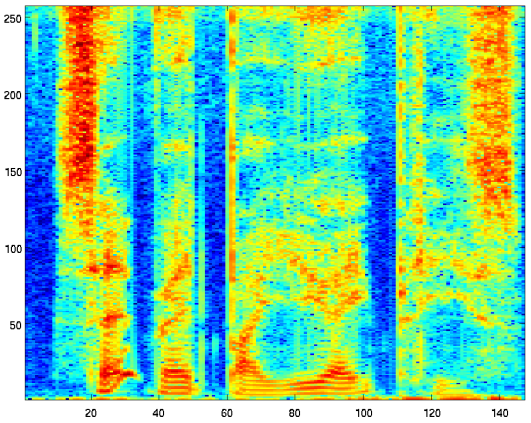
(e) Estimated SM



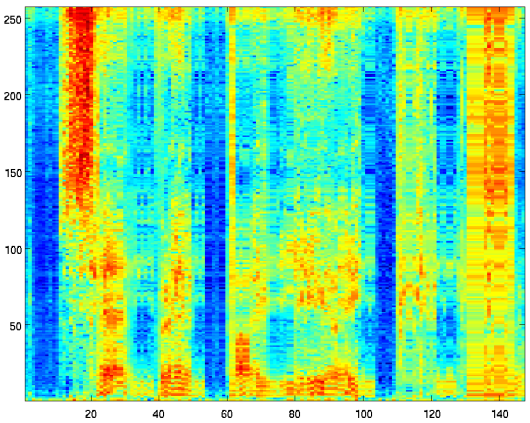
(f) Noise estimate $\hat{D}_S$



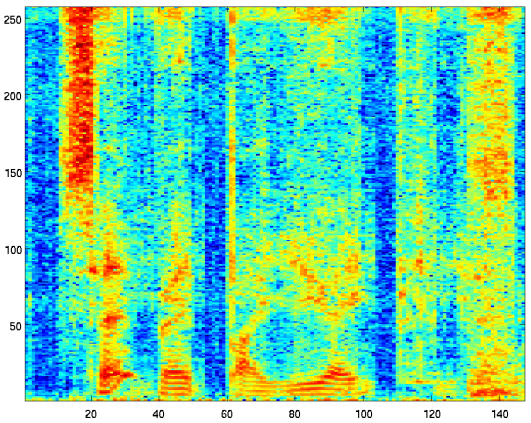
(g) Noisy spectrogram Y



(h) Clean speech spectrogram X

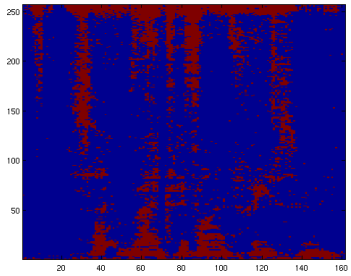


(i) Speech estimate $\hat{X}$

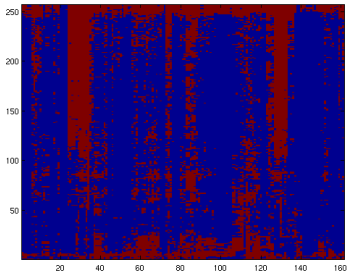


(j) SM speech estimate $\hat{X}_{SM}$

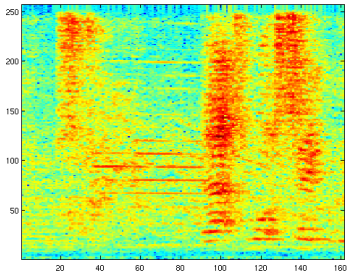
Figure B.6: Experiment 2: Figure showing an easy example SE setting for a 2-channel SPNHMM, File: srbu8p, Task: SDc, noise at +0dB, Models: <4><5>.



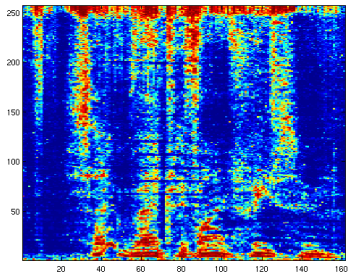
(a) IBM



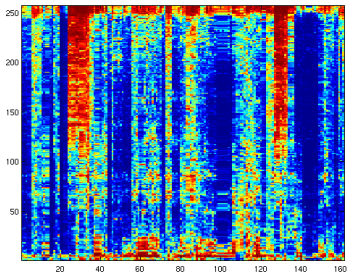
(b) Estimated BM



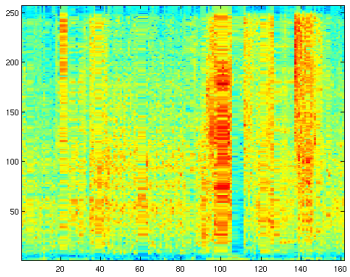
(c) Noise spectrogram D



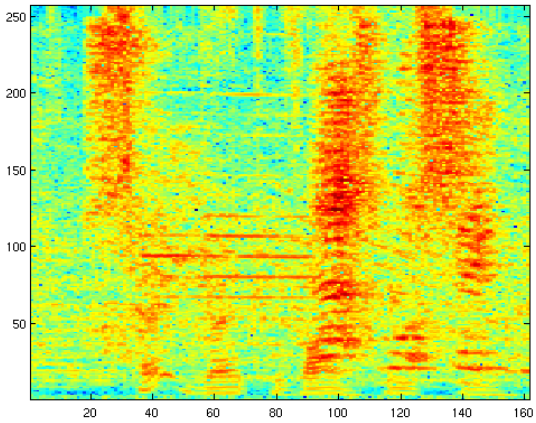
(d) ISM



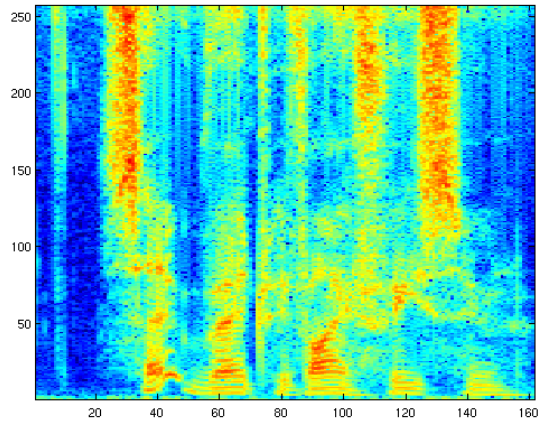
(e) Estimated SM



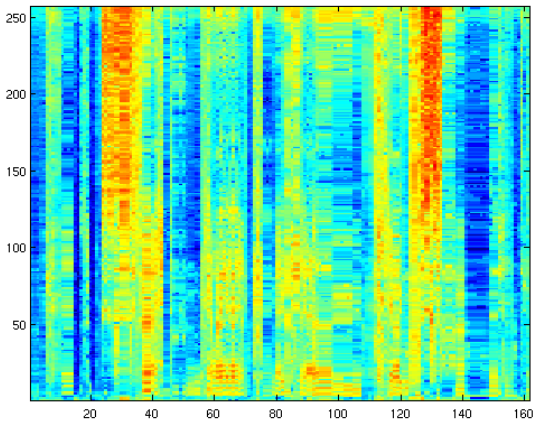
(f) Noise estimate $\hat{D}_S$



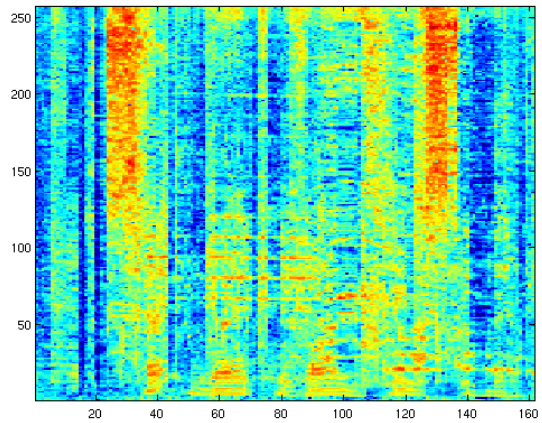
(g) Noisy spectrogram Y



(h) Clean speech spectrogram X

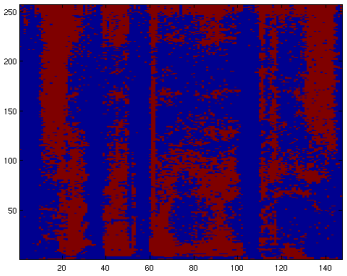


(i) Speech estimate $\hat{X}$

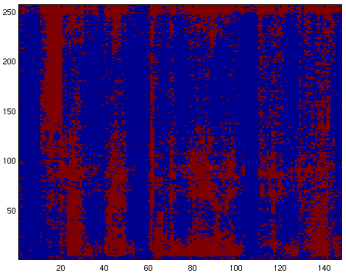


(j) SM speech estimate $\hat{X}_{SM}$

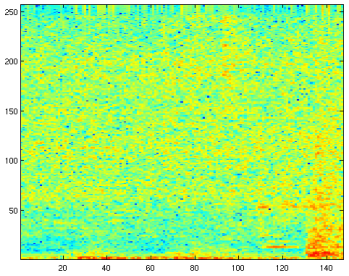
Figure B.7: Experiment 2: Figure showing a difficult example SE setting for a 2-channel SPNHMM, File: srwi3s, Task: SDc, noise at +0dB, Models: <4><5>.



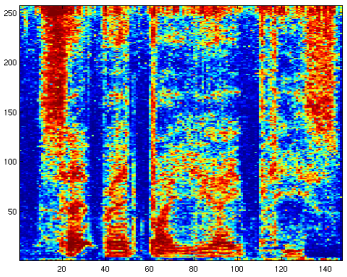
(a) IBM



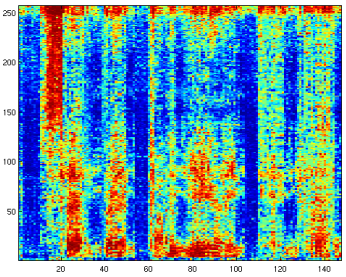
(b) Estimated BM



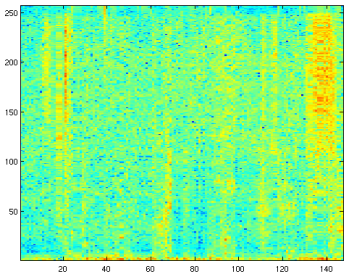
(c) Noise spectrogram D



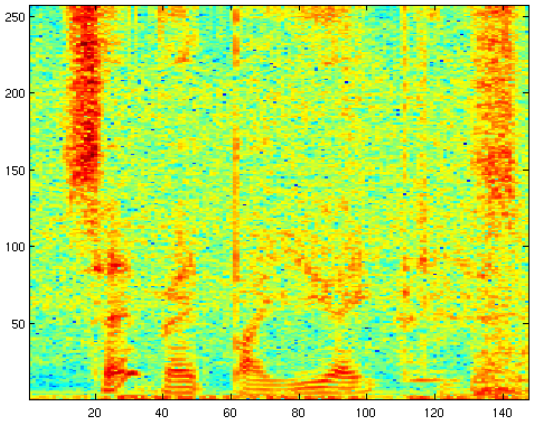
(d) ISM



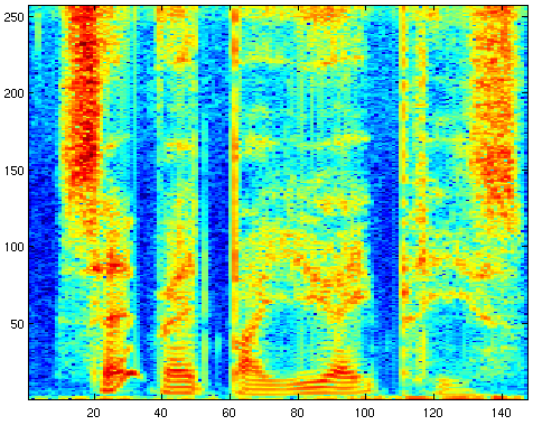
(e) Estimated SM



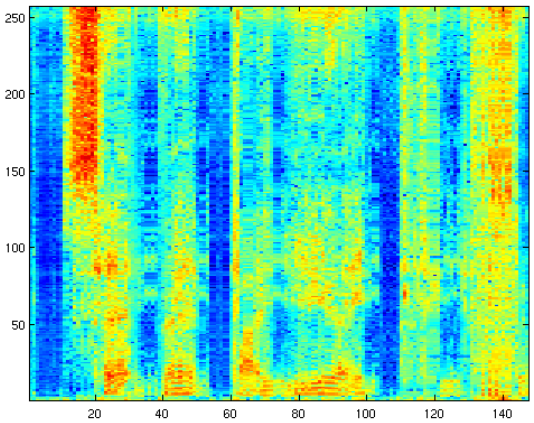
(f) Noise estimate $\hat{D}_S$



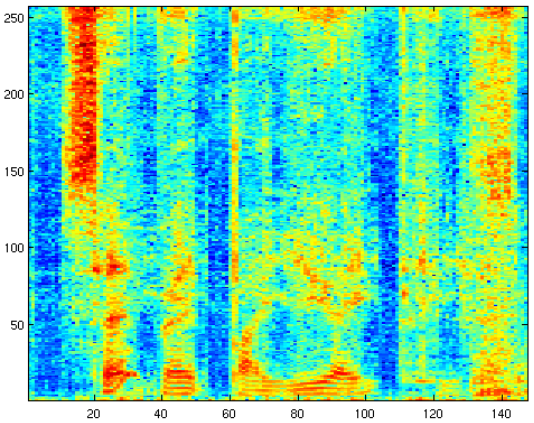
(g) Noisy spectrogram Y



(h) Clean speech spectrogram X

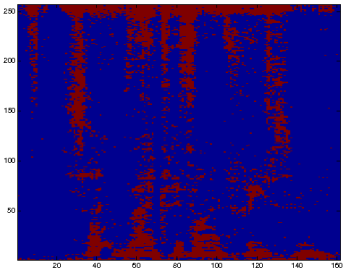


(i) Speech estimate $\hat{X}$

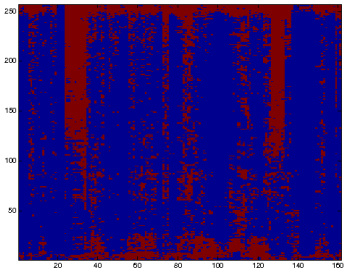


(j) SM speech estimate $\hat{X}_{SM}$

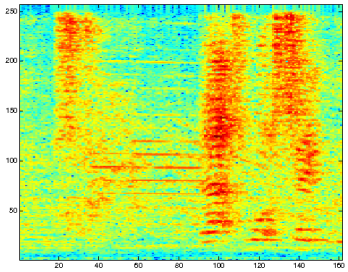
Figure B.8: Experiment 2: Figure showing an easy example SE setting for a 2-channel SPNHMM with cMPE, File: srbu8p, Task: SDc, noise at +0dB, Model: <4><5>.



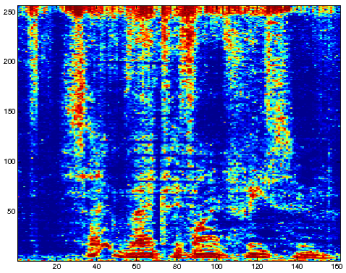
(a) IBM



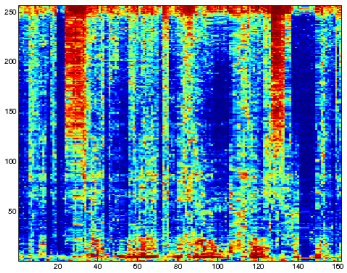
(b) Estimated BM



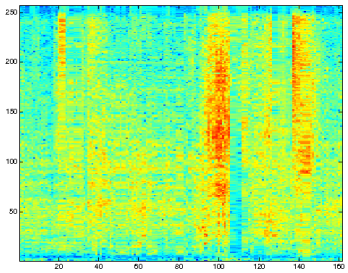
(c) Noise spectrogram D



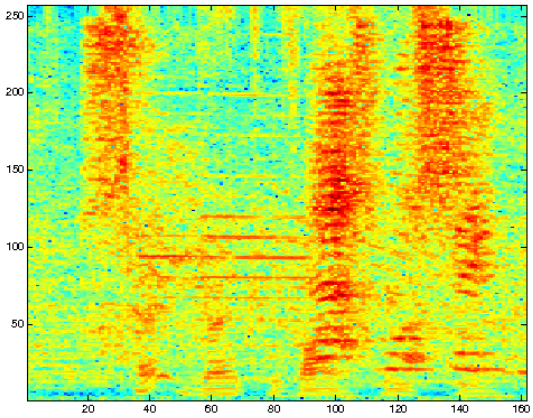
(d) ISM



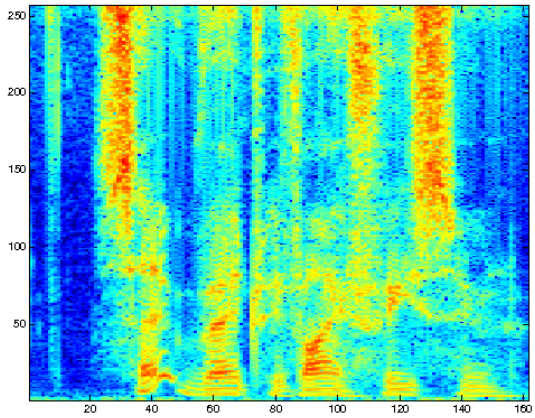
(e) Estimated SM



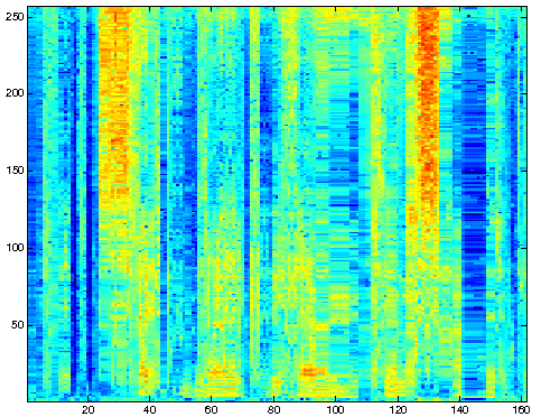
(f) Noise estimate $\hat{D}_S$



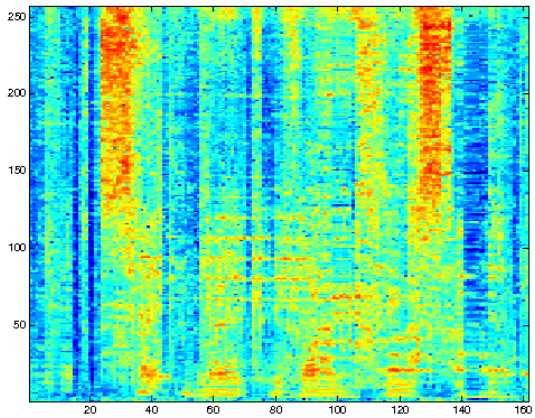
(g) Noisy spectrogram Y



(h) Clean speech spectrogram X



(i) Speech estimate $\hat{X}$



(j) SM speech estimate $\hat{X}_{SM}$

Figure B.9: Experiment 2: Figure showing a difficult example SE setting for a 2-channel SPNHMM with cMPE, File: srwi3s, Task: SDc, noise at +0dB, Model: <4><5>.

C Results

In this appendix, we provide detailed results of all experiments conducted. The symbols used in the plots are the same as Chapter 4 and denote the following models:

($*$) Noisy truth
($\times$) IMCRA reference
($\triangle$) SPNHMM, see Section 4.2.1
($\triangledown$) SPNHMM with cMPE, see Section 4.2.2
($\Diamond$) 2ch frame-wise SPN, see Section 4.3.1
($\triangleleft$) 2ch SPNHMM, see Section 4.3.2
($\triangleright$) 2ch SPNHMM with cMPE, see Section 4.3.3

C.1 Detailed results of Experiment 1

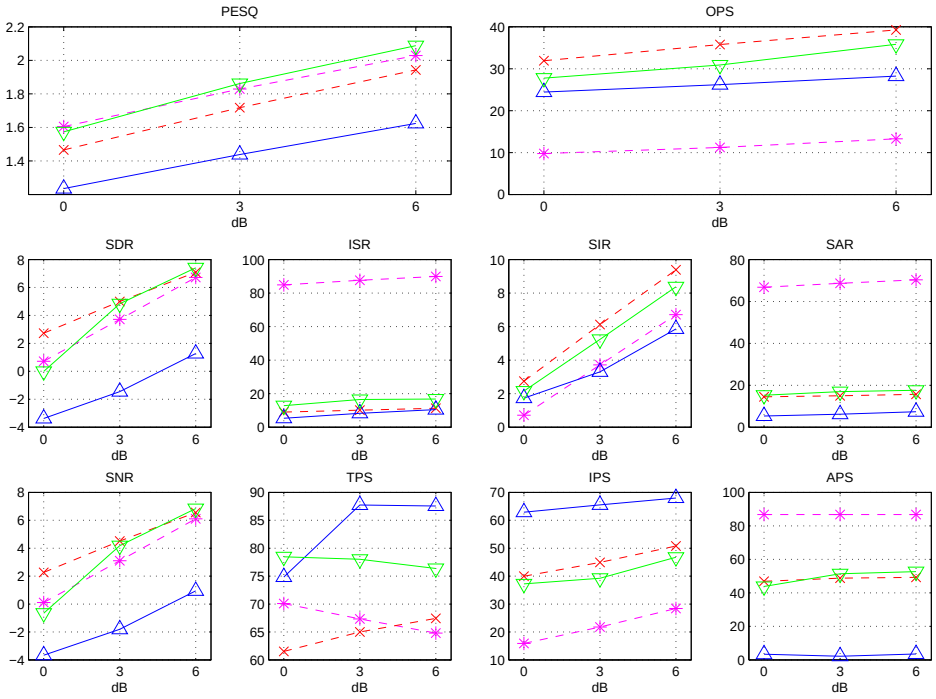


Figure C.1: Experiment 1: Detailed Results (SDc), DT-approach.

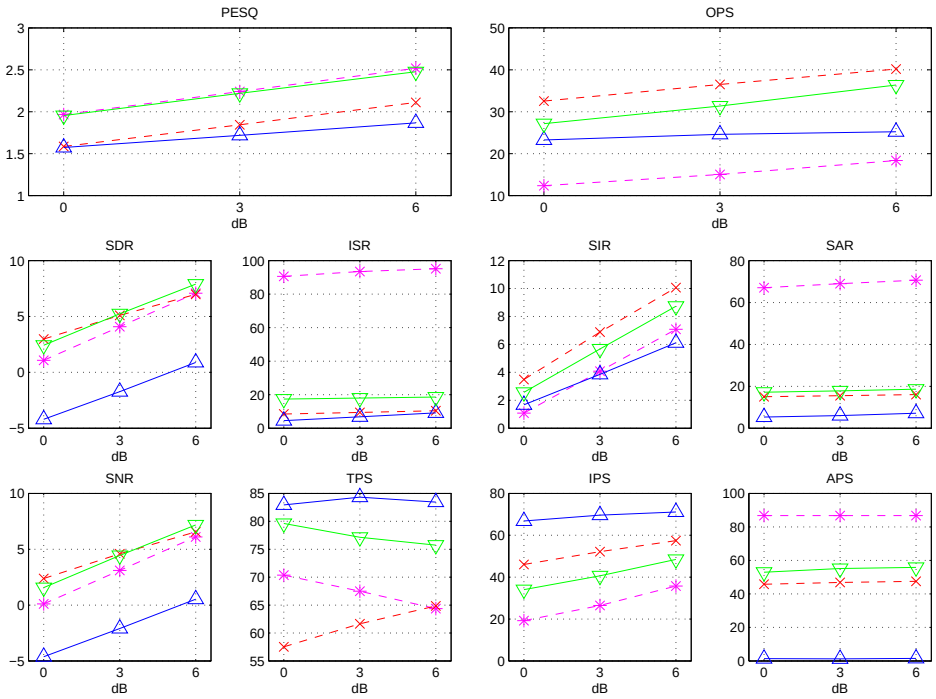


Figure C.2: Experiment 1: Detailed Results (SDr), DT-approach.

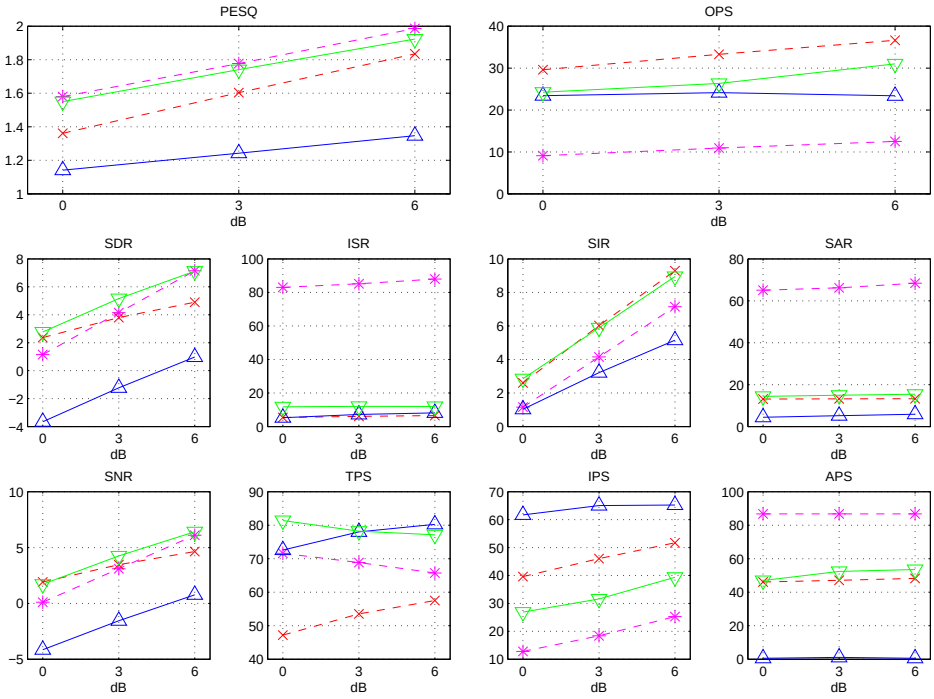


Figure C.3: Experiment 1: Detailed Results (SIc), DT-approach.

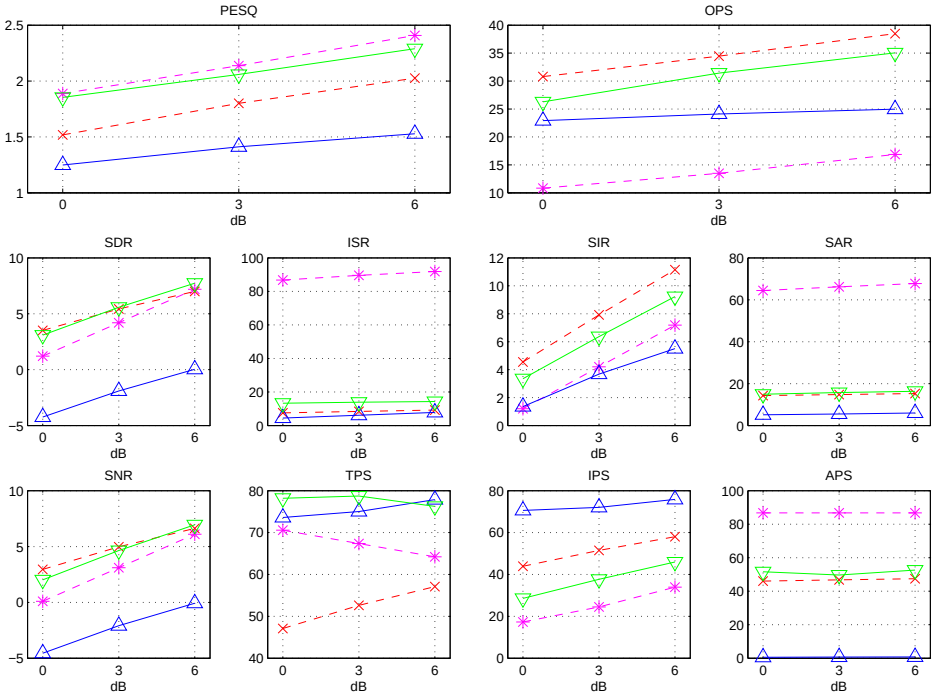


Figure C.4: Experiment 2: Detailed Results (SIr), DT-approach.

C.2 Detailed results of Experiment 2

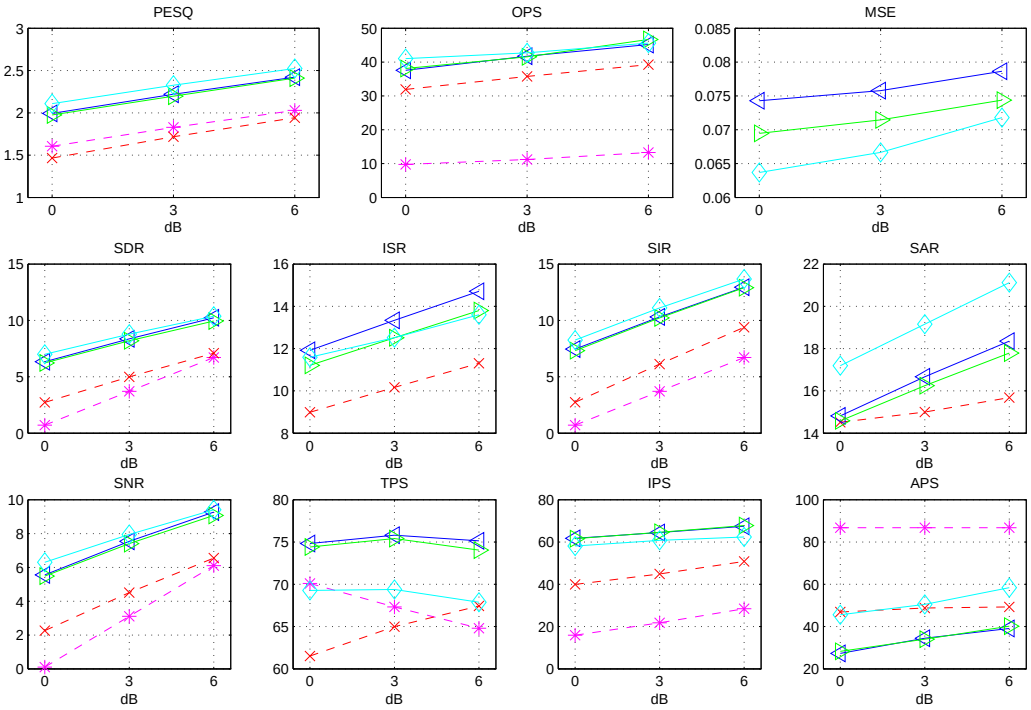


Figure C.5: Experiment 2: Detailed Results (SDc), IM-approach with a softmask.

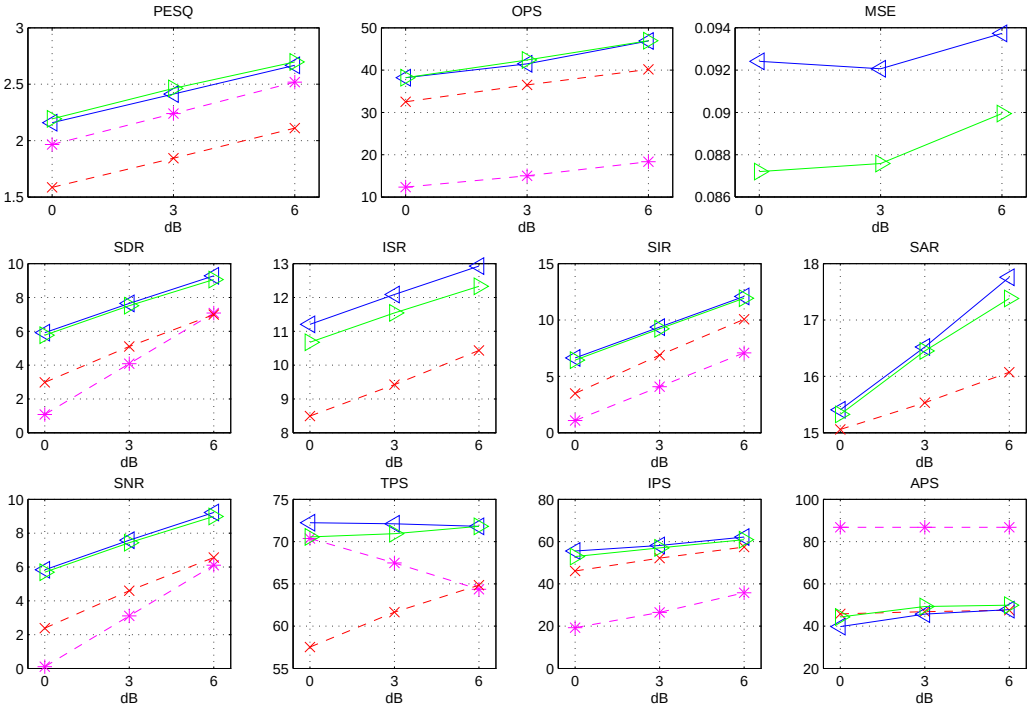


Figure C.6: Experiment 2: Detailed Results (SDr), IM-approach with a softmask.

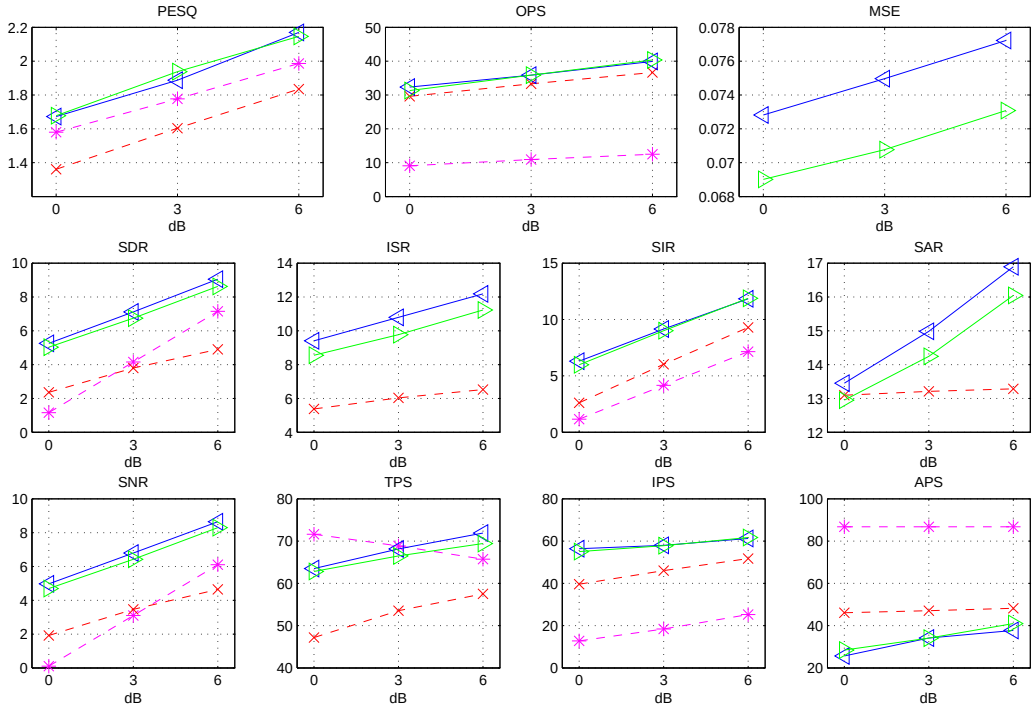


Figure C.7: Experiment 2: Detailed Results (SIc), IM-approach with a softmask.

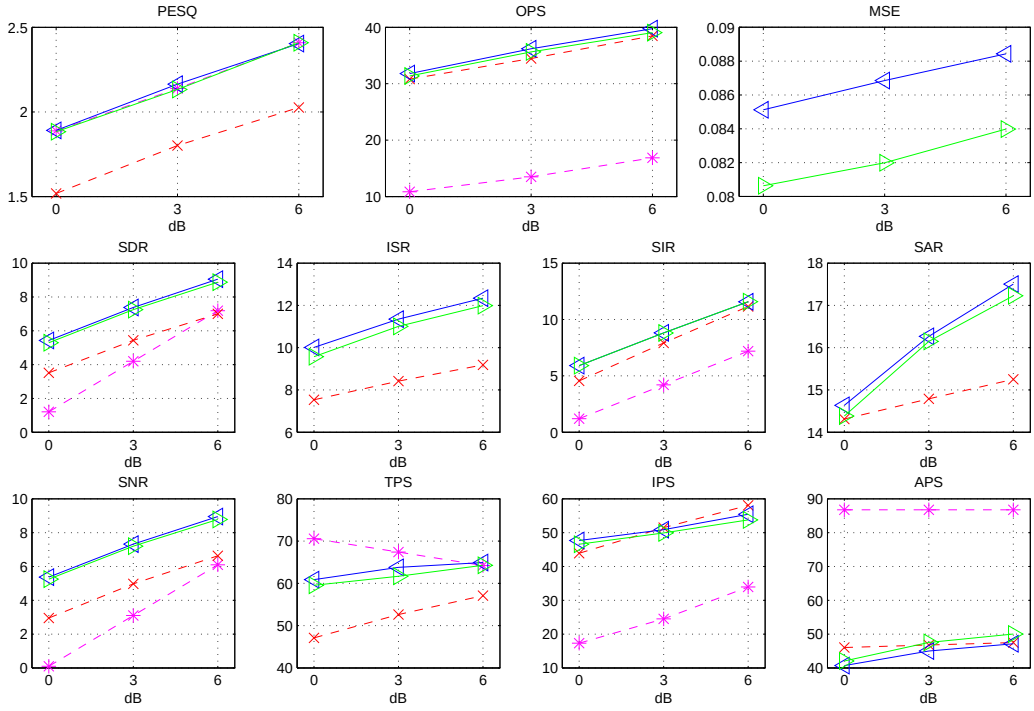


Figure C.8: Experiment 2: Detailed Results (SIr), IM-approach with a softmask.

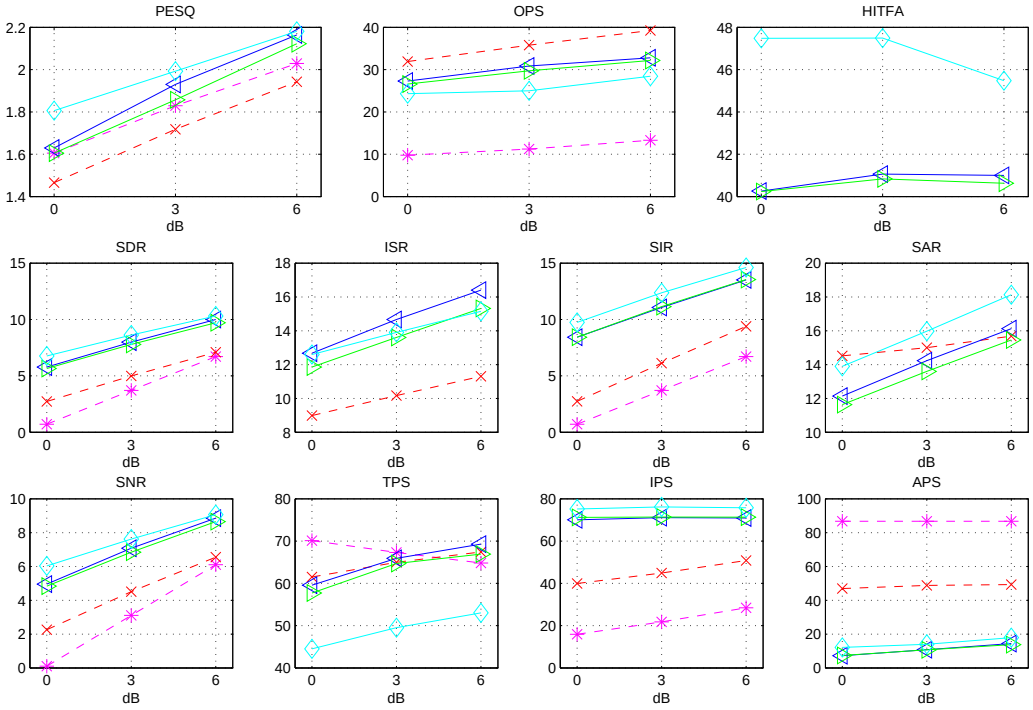


Figure C.9: Experiment 2: Detailed Results (SDc), IM-approach with a binary mask.

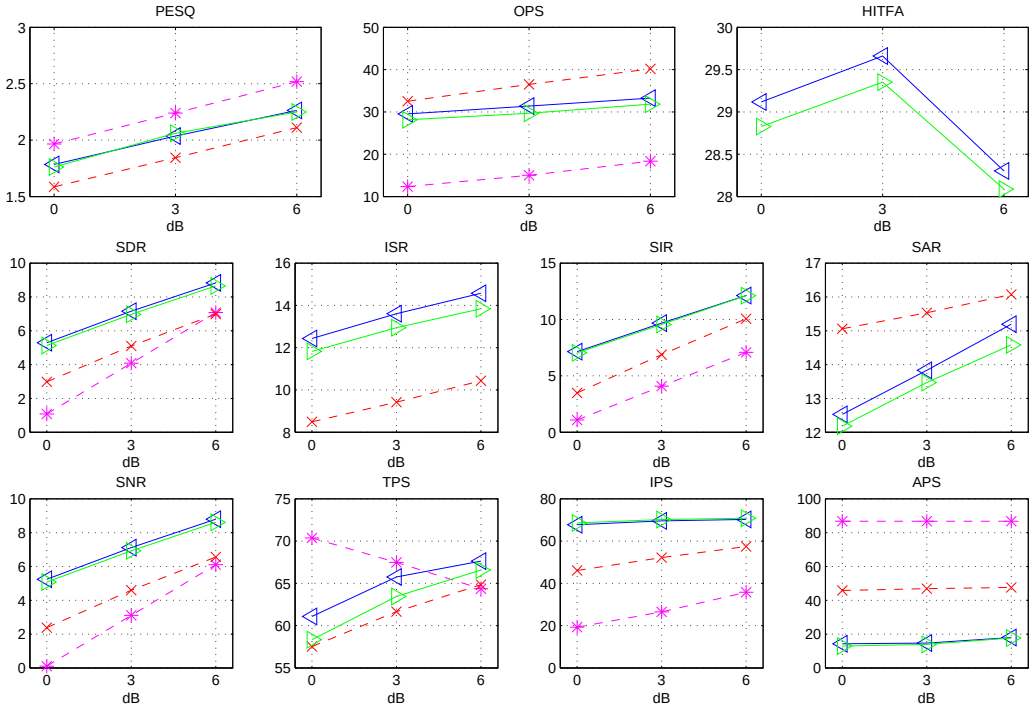


Figure C.10: Experiment 2: Detailed Results (SDr), IM-approach with a binary mask.

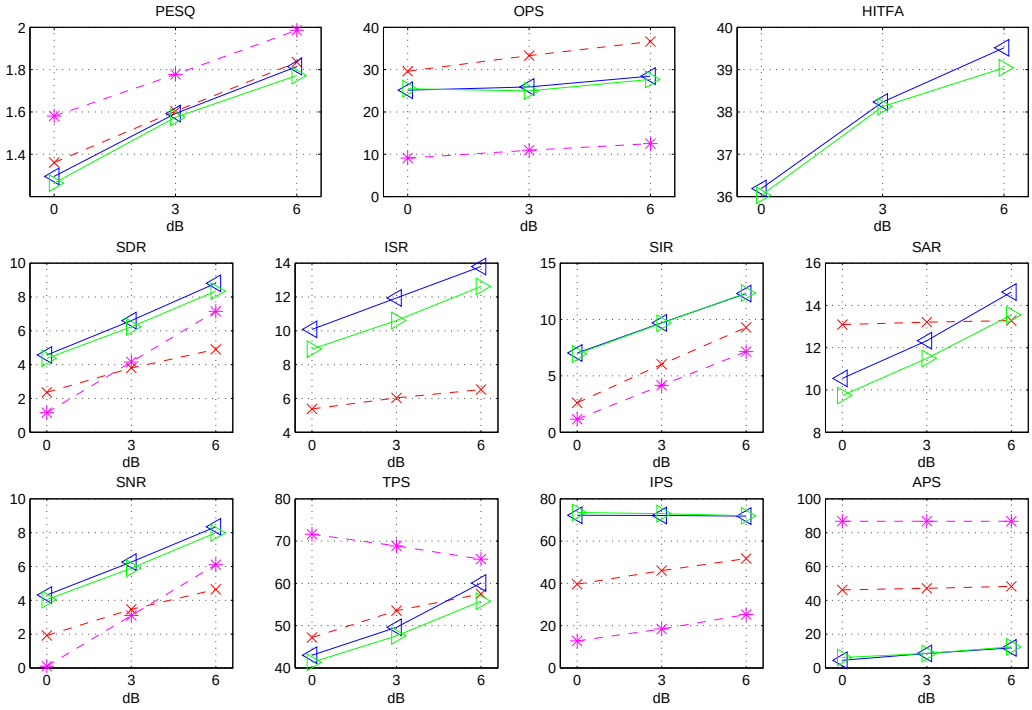


Figure C.11: Experiment 2: Detailed Results (SIc), IM-approach with a binary mask.

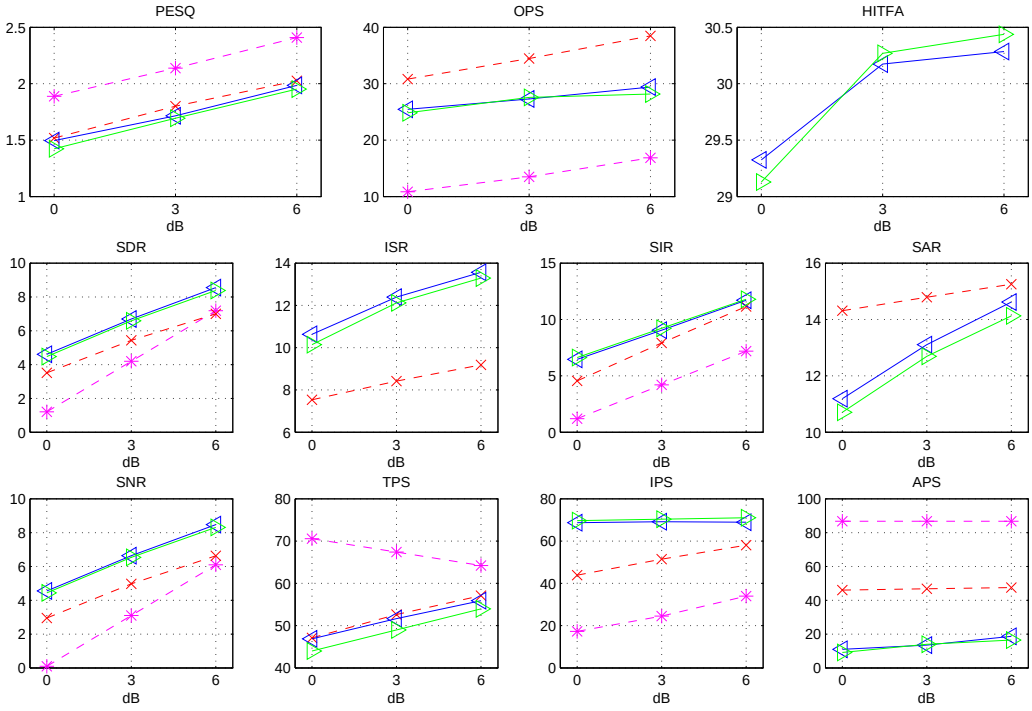


Figure C.12: Experiment 2: Detailed Results (SIr), IM-approach with a binary mask.

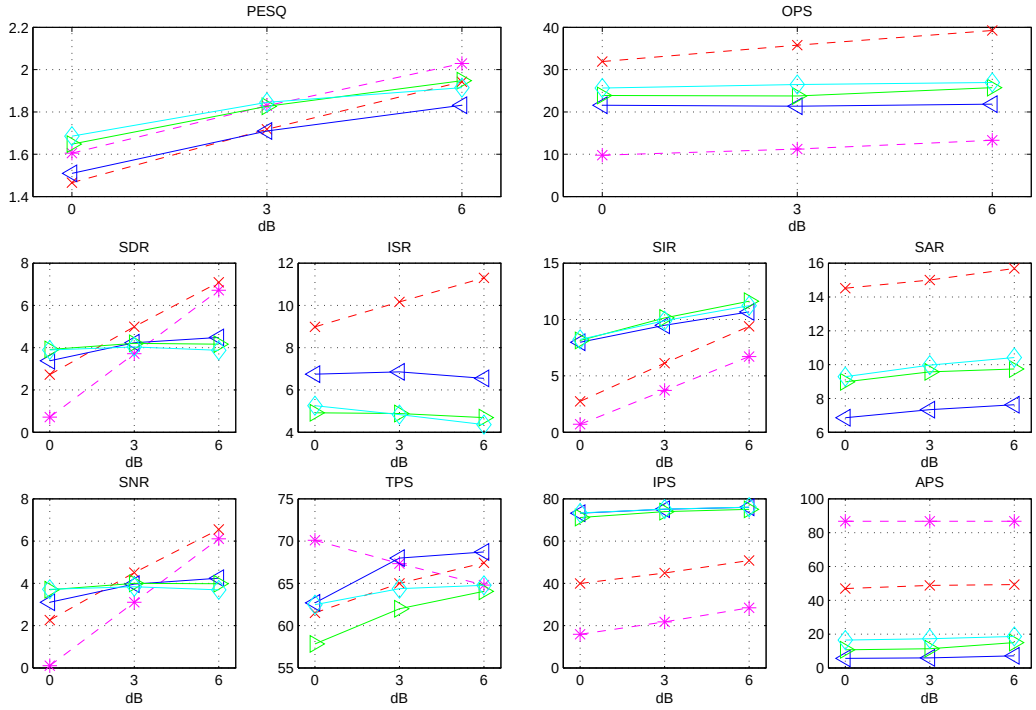


Figure C.13: Experiment 2: Detailed Results (SDc), DT-approach.

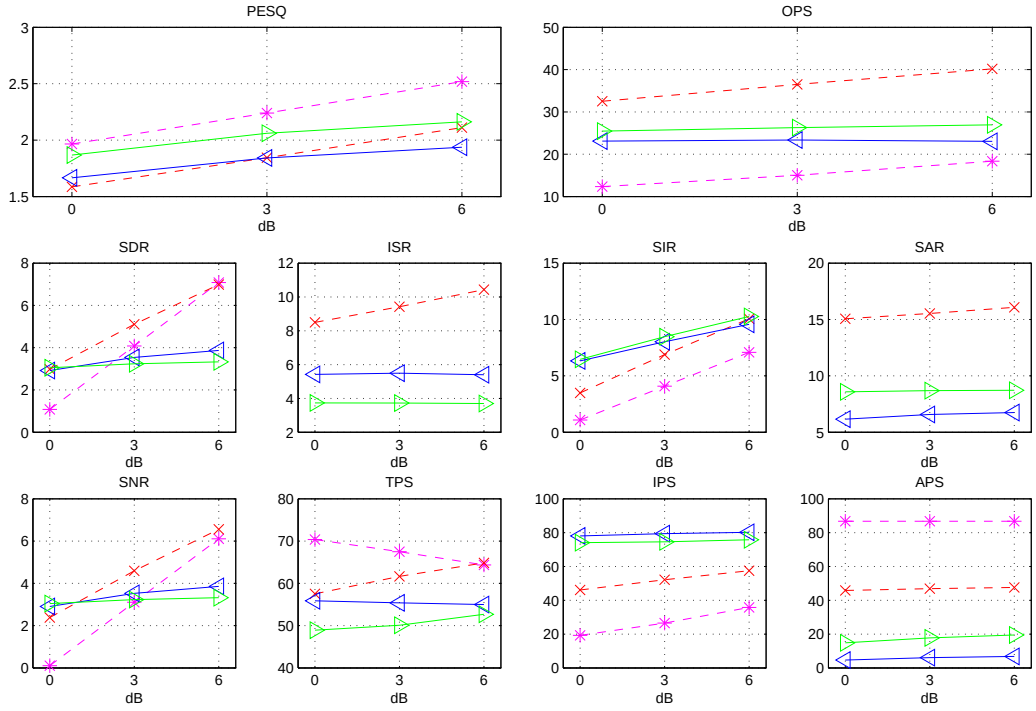


Figure C.14: Experiment 2: Detailed Results (SDr), DT-approach.

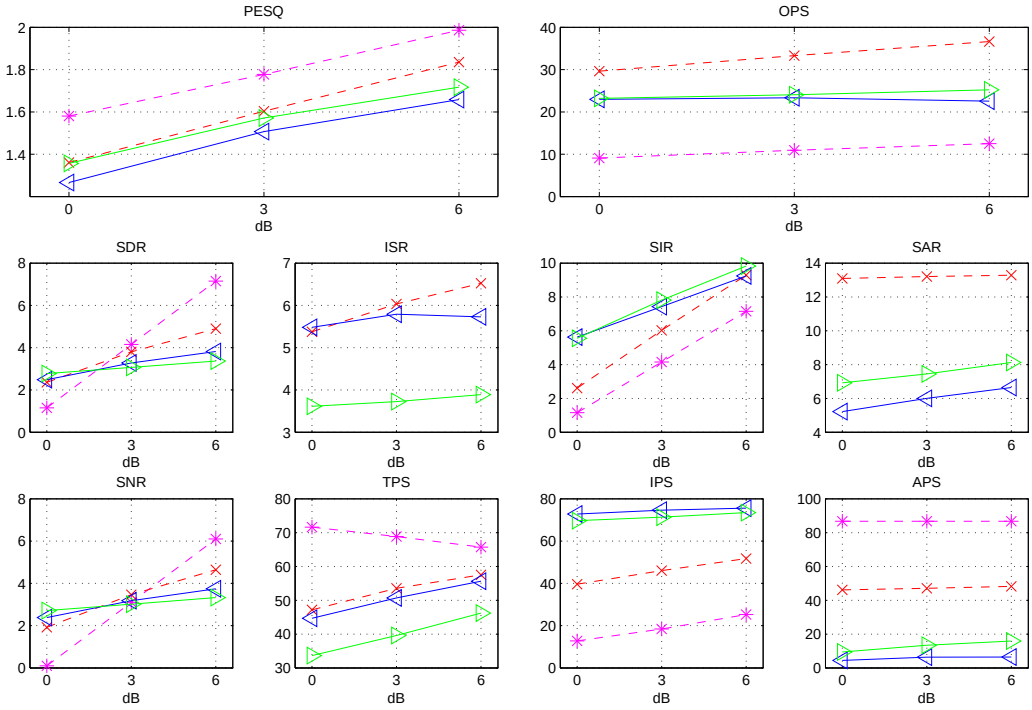


Figure C.15: Experiment 2: Detailed Results (SIc), DT-approach.

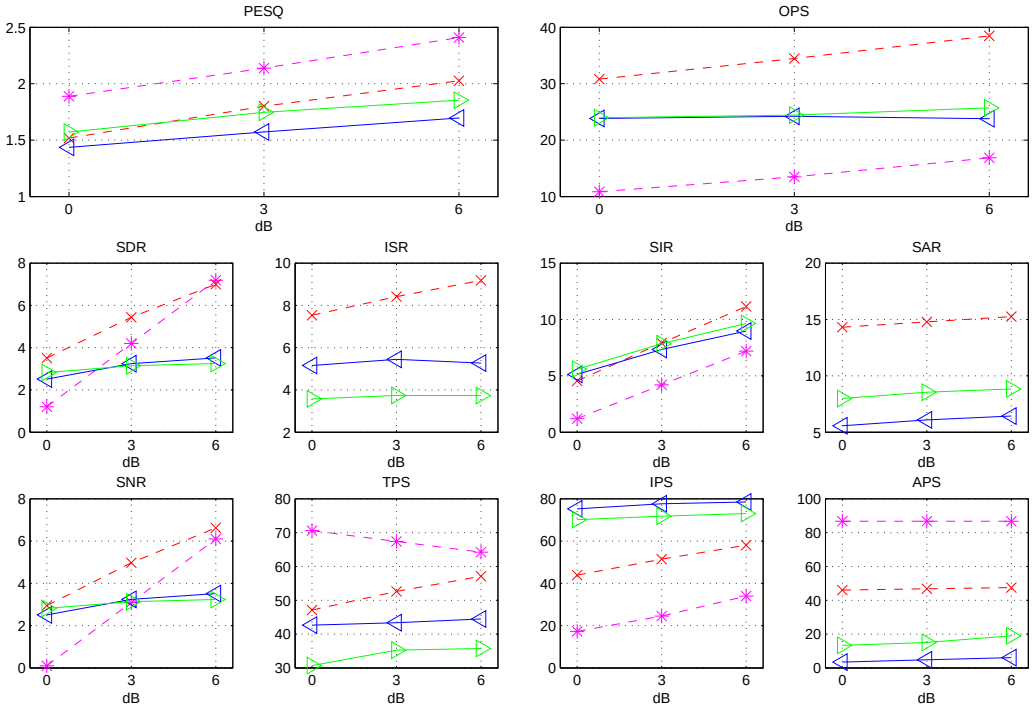


Figure C.16: Experiment 2: Detailed Results (SIr), DT-approach.

Bibliography

[1] P. C. Loizou, *Speech Enhancement: Theory and Practice*, 2nd ed. CRC, 2013.

[2] D. Wang, "On ideal binary mask as the computational goal of auditory scene analysis," in *Speech Separation by Humans and Machines*, 2005, pp. 181–197.

[3] S. T. Roweis, "One microphone source separation," in *Neural Information Processing Systems (NIPS)*, 2000, pp. 793–799.

[4] A. Ozerov, P. Philippe, R. Gribonval, and F. Bimbot, "One microphone singing voice separation using source-adapted models," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2005.

[5] J. R. Hershey, T. T. Kristjansson, S. Rennie, and P. A. Olsen, "Single channel speech separation using factorial dynamics," in *Neural Information Processing Systems (NIPS)*, 2006, pp. 593–600.

[6] M. Zöhrer and F. Pernkopf, "Representation models in single channel source separation," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015, submitted.

[7] S. T. Roweis, "Factorial models and refiltering for speech separation and denoising," in *Conference of the International Speech Communication Association (INTERSPEECH)*, 2003.

[8] R. Peharz, G. Kapeller, P. Mowlaee, and F. Pernkopf, "Modeling speech with sum-product networks: Application to bandwidth extension," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

[9] M. Zöhrer, R. Peharz, and F. Pernkopf, "On representation learning for artificial bandwith extension," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015, submitted.

[10] M. Zöhrer and F. Pernkopf, "General stochastic networks for classification," in *Neural Information Processing Systems (NIPS)*, 2014.

[11] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 113–120, 1979.

[12] R. McAulay and M. Malpass, "Speech enhancement using a soft-decision noise suppression filter," *IEEE Transactions on Acoustics, Speech and Signal Processing*, pp. 137–145, 1980.

[13] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, no. 6, pp. 1109–1121, 1984.

[14] R. Martin, "Noise power spectral density estimation based on optimal smoothing and minimum statistics," *IEEE Transactions on Speech and Audio Processing*, pp. 504–512, 2001.

[15] L. Lin, W. Holmes, and E. Ambikairajah, "Adaptive noise estimation algorithm for speech enhancement," *Electronics Letters*, pp. 754–755, 2003.

[16] H.-G. Hirsch and C. Ehrlicher, "Noise estimation techniques for robust speech recognition," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1995, pp. 153–156.

[17] I. Cohen, "Noise spectrum estimation in adverse environments: improved minima controlled recursive averaging." *IEEE Transactions on Speech and Audio Processing*, no. 5, pp. 466–475, 2003.

[18] Y. Ephraim and D. Malah, "Speech enhancement using a minimum mean-square error log-spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech and Signal Processing*, pp. 443–445, 1985.

[19] J. B. Boldt, "Binary masking and speech intelligibility," Ph.D. dissertation, Department of Electronic Systems, Aalborg University, 2011.

[20] R. Peharz and F. Pernkopf, "Exact maximum margin structure learning of bayesian networks," in *International Conference on Machine Learning (ICML)*, 2012.

[21] L. Benaroya, F. Bimbot, and R. Gribonval, "Audio source separation with a single sensor," *IEEE Transactions on Acoustics, Speech and Signal Processing*, pp. 191–199, 2006.

[22] W.-H. Tsai, D. Rodgers, and H.-M. Wang, "Blind clustering of popular music recordings based on singer voice characteristics," *Computer Music Journal*, pp. 68–78, 2004.

[23] M. Wohlmayr, M. Stark, and F. Pernkopf, "A probabilistic interaction model for multipitch tracking with factorial hidden markov models," *IEEE Transactions on Audio, Speech and Language Processing (ACM)*, pp. 799–810, 2011.

[24] J. R. Hershey, S. J. Rennie, P. A. Olsen, and T. T. Kristjansson, "Super-human multi-talker speech recognition: A graphical modeling approach," *Computer Speech and Language*, pp. 45–66, 2010.

[25] S. Rennie, J. Hershey, and P. Olsen, "Single channel multi-talker speech recognition: Graphical modeling approaches," *IEEE Signal Processing Magazine, Special Issue on Graphical Models*, 2010.

[26] G. E. Hinton and S. Osindero, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, p. 2006, 2006.

[27] G. E. Hinton, L. Deng, D. Yu, G. E. Dahl, A. rahman Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, pp. 82–97, 2012.

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

[29] Y. Wang and D. Wang, "Cocktail party processing via structured prediction," in *Neural Information Processing Systems (NIPS)*, P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., 2012, pp. 224–232.

[30] S. J. Rennie, P. Fousek, and P. L. Dognin, "Factorial hidden restricted boltzmann machines for noise robust speech recognition." in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2012, pp. 4297–4300.

[31] M. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013.

[32] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *Computing Research Repository (CoRR)*, vol. abs/1207.0580, 2012.

[33] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2001, pp. 749–752.

[34] V. Emiya, E. Vincent, N. Harlander, and V. Hohmann, "Subjective and objective quality assessment of audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, pp. 2046–2057, 2011.

[35] R. Huber and B. Kollmeier, "PEMO-Q - a new method for objective audio quality assessment using a model of auditory perception," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 6, pp. 1902–1911, 2006.

[36] K. Brandenburg, "Mp3 and aac explained," in *Audio Engineering Society International Conference on High Quality Audio Coding*, 1999.

[37] H. Poon and P. Domingos, "Sum-product networks: A new deep architecture," in *Uncertainty in Artificial Intelligence*, 2011, pp. 337–346.

[38] A. Darwiche, "A Differential Approach to Inference in Bayesian Networks," *IEEE Transactions on Audio, Speech and Language Processing (ACM)*, vol. 50, no. 3, pp. 280–305, 2003.

[39] R. Peharz, B. Geiger, and F. Pernkopf, "Greedy Part-Wise Learning of Sum-Product Networks," in *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, vol. 8189. Springer Berlin, 2013, pp. 612–627.

[40] H. Poon and P. Domingos, "http://alchemy.cs.washington.edu/spn/," 2011, (online).

[41] F. Samaria and A. Harter, "Parameterisation of a stochastic model for human face identification," in *IEEE Workshop on Applications of Computer Vision*, 1994, pp. 138–142.

[42] A. Dennis and D. Ventura, "Learning the architecture of sum-product networks using clustering on variables," in *Neural Information Processing Systems (NIPS)*, 2012, pp. 2042–2050.

[43] R. Gens and P. Domingos, "Learning the Structure of Sum-Product Networks," in *International Conference on Machine Learning (ICML)*, 2013, pp. 873–880.

[44] ——, "Discriminative learning of sum-product networks," in *Neural Information Processing Systems (NIPS)*, 2012, pp. 3248–3256.

[45] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems 19*, 2007, pp. 153–160.

[46] E. Vincent, J. Barker, S. Watanabe, J. L. Roux, F. Nesta, and M. Matassoni, "The second 'chime' speech separation and recognition challenge: An overview of challenge systems and outcomes." in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2013, pp. 162–167.

[47] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Transaction on Communication*, vol. 28, no. 1, pp. 84–95, 1980.

[48] I. Cohen and S. Gannot, "Spectral enhancement methods," in *Springer Handbook of Speech Processing*, J. Benesty, M. Sondhi, and Y. Huang, Eds. Springer Berlin Heidelberg, 2008, pp. 873–902.

[49] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, 1st ed. Springer, Oct. 2007.

[50] R. Peharz, R. Gens, and P. Domingos, "Learning selective sum-product networks," in *International Conference on Machine Learning (ICML)*, 2014.

[51] M. Cookeand, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 120, no. 5, pp. 2421–2424, 2006.