Thesis n° 10 908

Modelling cochlea and its interaction with the auditory path for speech processing

Presented on 29th July 2025

School of Engineering
Medical Image Processing Laboratory
Doctoral program in Electrical Engineering

for the award of the degree of Docteur ès Sciences (PhD)

by

Louise Clothilde COPPIETERS DE GIBSON

Accepted on the jury's recommendation

Prof. J.-Ph. Thiran, jury president
Prof. D. N. A. Van De Ville, Dr Ph. N. Garner, thesis directors
Dr Y. Su, examiner
Prof. D. Kolossa, examiner
Prof. S.-C. Liu, examiner

École
polytechnique
fédérale
de Lausanne

2025

A Scout smiles and whistles under all circumstances
— Baden-Powell

To my beloved husband, my family and all friends who supported me during this journey …

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Phil Garner, for your availability, support, and guidance throughout this four-year journey. Your presence and dedication at every stage of this thesis made a significant difference, and I am sincerely grateful for the insight, patience, and encouragement you provided.

I would also like to thank the members of the jury for your valuable feedback and constructive suggestions, which have greatly contributed to refining and strengthening this work.

To the friends I've made along the way — thanks to you, Idiap was not only a workplace, but truly became a second family. Thank you for the shared moments, the laughs, the countless cakes at lunchtime, the kicker matches, and so much more. To those who were crazy enough to try ski touring or mountain climbing with me: I hope our adventures continue even beyond the PhD.

To the friends I've known before — thank you for always being there, for reminding me who I am outside of research, and for your support from both my homeland and Switzerland.

To the flatmates who shared this journey with me — through cooking, playing board games, and spending countless evenings reinventing the world — Evann, Benoît, and Jonathan — this experience would not have been the same without you.

To my parents, brothers, and sisters — despite the distance, your love and support have always felt close. Thank you for your visits, your encouragement, and for believing in me from afar.

To my family-in-law, thank you for the warm welcome you gave me in this new country and for your support during these past five years.

And finally, to my husband — your love, patience, and steadfast presence have been my greatest source of strength. Thank you for your support, for your incredible belief in my abilities, and for simply being yourself. You pulled me out of my comfort zone and into a world of snow-covered summits, 4000-meter climbs, and winter nights in mountain huts or under a tent on a glacier. You've added life and adventure to every moment we've shared.

Abstract

This thesis explores the intersection of physiological modelling and computational techniques in advancing Automatic Speech Recognition (ASR) systems. Contemporary ASR, often driven by attention models and self-supervised learning, has achieved remarkable accuracy, but remains decoupled from more recent physiological principles. In the meantime, significant progress has been made in understanding the function of the cochlea, the auditory system's sensory organ. Originally viewed as a passive filter bank, the cochlea is now understood to function as an active amplifier, well modelled by a Hopf oscillator.

The goal of this thesis is to investigate how the latest understanding of physiology can be combined and studied within deep learning based ASR models. To this end, the thesis is organised as two interacting threads.

In a first thread, we investigate modularity, which proposes strategies to integrate and combine different types of machine learning models, using different experts, or combine new frontend models with pretrained large transformer models. In a preliminary study, we show that modularity can be used to optimise an ASR model for different types of environmental noise.

In a second thread, we utilise modularity to investigate how to incorporate improved cochlear understanding into ASR systems, creating a two-way bridge where insights from computational approaches inform auditory physiology. After studying established techniques such as CARFAC and SincNet, we investigate trainable filter banks within a convolutional neural network (CNN) structure to determine key hyperparameters for ASR performance. This study also highlights interesting insights filters tend to learn when able to train in an ASR context.

Finally, we combine the threads by embedding a Hopf-based cochlear model within an ASR system, informed by the learned filter bank parameters. We show that the Hopf mechanism demonstrates the expected cube root compression and gain control. Moreover, a larger feedback loop, modelling the olivocochlear efferent path further enhances the overall performance.

The resulting system, offers valuable insights for future interdisciplinary studies between ASR and physiological auditory models.

Key words: ASR, cochlear model, Hopf oscillator, SincNet, efferent pathway, active amplification mechanism, CARFAC, self-supervised models

Résumé

Cette thèse explore l'intersection entre la modélisation physiologique et les techniques computationnelles pour l'amélioration des systèmes d' *Automatic Speech recognition* (ASR). Les systèmes ASR contemporains, souvent basés sur des modèles d'attention et l'apprentissage auto-supervisé, ont atteint une précision remarquable, mais restent déconnectés des principes physiologiques les plus récents.

En parallèle, des avancées significatives ont été réalisées dans la compréhension du fonctionnement de la cochlée, l'organe sensoriel du système auditif. Longtemps considérée comme un simple banc de filtres passifs, la cochlée est désormais reconnue comme un amplificateur actif, modélisé par un oscillateur de Hopf.

L'objectif de cette thèse est d'investiguer comment ces avancées dans la compréhension physiologique peuvent être étudiées en combinaison avec l'état de l'art des techniques d'ASR.

Pour ce faire, la thèse est organisée en deux axes. Dans un premier axe, nous investiguons la modularité, qui propose des stratégies pour intégrer et combiner différents types de modèles de *deep learning*, en utilisant différents experts, ou combiner de nouveaux modèles de prétraitement acoustique avec des grands modèles de transformeurs pré-entraînés. Dans une étude préliminaire, nous montrons que la modularité peut être utilisée pour optimiser un modèle d'ASR face à différents types de bruits environnementaux.

Dans un second axe, nous exploitons la modularité pour explorer comment intégrer une meilleure compréhension de la cochlée dans les systèmes ASR, créant ainsi un lien bidirectionnel où les avancées des approches computationnelles peuvent à leur tour inspirer la recherche dans la compréhension de la physiologie auditive. Après avoir étudié des techniques établies telles que CARFAC et SincNet, nous analysons ce qu'apprennent des bancs de filtres entraînables au sein d'une architecture d'un réseau de neurones convolutifs (CNN) afin de déterminer les hyperparamètres clés qui pourront être transposés à des modèles cochléaires plus complexes. Cette étude met en évidence des observations intéressantes sur les filtres que le modèle tend à apprendre lorsqu'ils sont entraînés dans un contexte ASR.

Enfin, nous réunissons ces deux axes en intégrant un modèle cochléaire basé sur les oscillateurs de Hopf au sein d'un système ASR, en nous appuyant sur les paramètres appris par les bancs de filtres. Nous montrons que le mécanisme de Hopf reproduit la compression en racine cubique ainsi que le contrôle de gain attendus. De plus, une boucle de rétroaction plus large, inspirée de la voie efférente olivocochléaire, améliore davantage les performances globales. Le système résultant offre des perspectives intéressantes pour de futures études interdisciplinaires entre l'ASR et les modèles physiologiques de l'audition.

Mots clefs : ASR, modèle cochléaire, oscillateur de Hopf, SincNet, voie efférente, méchanisme d'amplification actif, CARFAC, modèles d'auto-entraînement

Contents

Abstract (English/Français) iii

List of figures xi

List of tables xv

Glossary xvii

1 Introduction 1
- 1.1 Context and Motivation 1
- 1.2 Problem statement, research objectives and limitations 3
- 1.3 Outline and main contributions 4

2 Background 7
- 2.1 The cochlea 7
  - 2.1.1 Anatomy and tonotopic organisation 8
  - 2.1.2 High-level working of the cochlea 9
  - 2.1.3 The active amplification mechanism 10
  - 2.1.4 Feedback connections in auditory path. 11
- 2.2 Broad overview of cochlear Models 12
  - 2.2.1 Gammatone filter 13
  - 2.2.2 Neurotransmitter based and spiking models 13
  - 2.2.3 Basilar membrane coupled to fluid dynamics 13
  - 2.2.4 Hair cell interaction mechanism 14
- 2.3 ASR 16
  - 2.3.1 Evolution of ASR 16
  - 2.3.2 ASR structure 18
  - 2.3.3 Speech components and feature extraction 18
  - 2.3.4 Databases 19
  - 2.3.5 Metric 22

3 Modularity 23
- 3.1 Introduction 24
- 3.2 Background 25
  - 3.2.1 Modular networks 25

3.2.2 Conformer . . . 26
3.3 Method . . . 26
3.3.1 Fixed routing . . . 26
3.3.2 Learned routing . . . 28
3.4 Experiments . . . 28
3.4.1 Dataset . . . 28
3.4.2 Baseline and framework . . . 28
3.4.3 Fixed Routing . . . 29
3.4.4 Learned routing . . . 30
3.5 Conclusion . . . 33

4 Hopf oscillator 35
4.1 Background . . . 36
4.1.1 Types of oscillators . . . 36
4.1.2 Cochlear model equations in the literature . . . 39
4.1.3 Criticality concept . . . 44
4.1.4 Take aways for building our model . . . 45
4.2 Bifurcations . . . 46
4.2.1 Fold bifurcation . . . 46
4.2.2 Transcritical bifurcation . . . 47
4.2.3 Pitchfork bifurcation . . . 48
4.2.4 Hysteresis bifuration . . . 49
4.2.5 The Hopf oscillator with external input: a combination of hysteresis and pitchfork bifurcation . . . 50
4.3 Simulation of oscillators . . . 51
4.3.1 A single oscillator . . . 51
4.3.2 Multiple oscillators . . . 52
4.3.3 Experiment on an artificially built signal . . . 54
4.3.4 Experiment on an audio signal . . . 57
4.3.5 Limitations of the simulation model . . . 59
4.4 Precomputed plausible cochlear features in an ASR system . . . 59
4.5 Conclusion . . . 60

5 Trainable filters 63
5.1 Introduction . . . 64
5.2 Background . . . 65
5.3 Initial Analysis . . . 66
5.3.1 SincNet setup . . . 66
5.3.2 Method . . . 66
5.3.3 Baseline . . . 67
5.3.4 Number of filters . . . 68
5.3.5 Scale after training . . . 69
5.3.6 Corollary . . . 70

5.4 Wide-band filter analysis . . . 70
  5.4.1 Wide-band initialization . . . 70
  5.4.2 Why wide-band filters? . . . 71
5.5 Conclusion . . . 73

6 Trainable filters with self-supervised pretrained model 75
6.1 Introduction . . . 77
6.2 Background . . . 78
  6.2.1 Self-Supervised Models . . . 78
  6.2.2 Cochlear Models . . . 80
  6.2.3 ASR with Trainable Filters . . . 81
  6.2.4 Speech Features . . . 82
6.3 Method . . . 82
  6.3.1 Overall Hypothesis . . . 82
  6.3.2 Pre-Trained Model . . . 83
  6.3.3 Experimental setup and training protocol . . . 83
  6.3.4 Dataset . . . 83
6.4 Experiments . . . 84
  6.4.1 Can Trainable Filters Replace the Encoder CNN in an Already Pre-Trained Model? . . . 84
  6.4.2 Does a Physiologically Adapted Front-End Perform as Well as a CNN in a Pre-Trained Model? . . . 89
  6.4.3 Can Trainable Filters be Incorporated during Self-Supervision? . . . 91
  6.4.4 Do Wide-Band Filters Appear in Some Other Training or Model Configurations? . . . 93
6.5 Conclusions . . . 96

7 Integration of Hopf oscillator module into an ASR system 99
7.1 Background . . . 100
7.2 Learning parameters . . . 100
7.3 Hopf oscillator module . . . 101
  7.3.1 Mathematical formulation . . . 101
  7.3.2 Implementation of the Hopf module . . . 102
  7.3.3 Characterization of the Hopf Module . . . 103
7.4 Integration in ASR . . . 111
  7.4.1 Challenges in bigger scale network . . . 111
  7.4.2 Adapting time constant of active amplification loop . . . 112
  7.4.3 Method . . . 113
  7.4.4 Integration of Hopf-module in a simple ASR structure . . . 113
  7.4.5 Impact of adding recurrence on the CNN frontend . . . 116
  7.4.6 Adding the efferent path feedback loop . . . 118
  7.4.7 A more complex implementation . . . 120
7.5 Noise addition . . . 121

7.6 Hardware implementations of Hopf oscillators 123
7.7 Conclusion 123

8 Conclusion 125
8.1 Further research and recommendations 126

A An appendix 129
A.1 Active force: a set of two differential equations 129
A.2 Equations for simulation of Hopf oscillator 130
A.2.1 Single oscillator with adaptive frequency 130
A.2.2 A series of Hopf oscillators 131

Bibliography 133

Curriculum Vitae 145

List of Figures

1.1 Overview of the research strategy. Inspired by the human auditory system, this study explores a plausible cochlear model integrated into a neural network, aiming to provide insights into both hearing technology and human physiology. 2
1.2 General overview of the thesis. 4

2.1 General overview of the thesis highlighting the background section. 7
2.2 Schematic of the human ear divided into outer, middle and inner ear. 8
2.3 Schematic of the organ of Corti 8
2.4 Schematic of the cochlear duct, the central membrane is an approximation of the frequency-selective basilar membrane. The high frequencies are detected on the beginning of the cochlear duct and the lower frequencies are detected at the end, near the apex. 9
2.5 Schematic of the working of the cochlea 10
2.6 Auditory path from cochlea to midbrain with the efferent paths from the auditory cortex and mibrain. 11
2.7 Schematic of CARFAC from Lyon (2017a). The y-components represent the state of the basilar membrane and the r-components represent the signals sent to the brain. 15
2.8 Structure of ASR system. 18
2.9 Frequency rates of different speech components. 19
2.10 Summary of the different noises of the CHiME4 dataset. 20

3.1 General overview of the thesis. 24
3.2 Fixed routing architecture 27
3.3 Learned routing architecture 27
3.4 Confusion matrices for the different number of experts: (a) for 2 experts, (b) 3 experts and (c) 5 experts. The x-axis represent the type of domain we have at the input and the y-axis the output of the network. 31
3.5 Loss function of different experiments: baseline and fixed and learned routing. 32

4.1 General overview of the thesis. 35
4.2 Schematic of the solutions of equation 4.5: the radius obtained in function of the bifurcation parameter (μ). The bifurcation occurs at $\mu = 0$. 38
4.3 Damping regime 38

4.4 Hopf bifurcation regime 39

4.5 Active amplification regime 39

4.6 Oscillator filterbank after frequency adaptation. 46

4.7 Evolution or $\dot{r}$ in function of r for different values of F. The solutions of the differential equation vary between 0,1 and 2 solutions. 47

4.8 Schematic of the solutions on the r-F plane. The stable and unstable nullclines are indicated in blue and red respectively. 47

4.9 Evolution or $\dot{r}$ in function of r for different values of μ. The solutions of the differential equation vary between 1 and 2 solutions. 47

4.10 Schematic of the solutions on the r-μ plane. The stable and unstable nullclines are indicated in blue and red respectively. 47

4.11 Evolution or $\dot{r}$ in function of r for different values of μ. The solutions of the differential equation vary between 1 and 3 solutions. 48

4.12 Schematic of the solutions on the r-μ plane. The stable and unstable nullclines are indicated in blue and red respectively. 48

4.13 Evolution or $\dot{r}$ in function of r for different values of F. The solutions of the differential equation vary between 1,2 and 3 solutions. 49

4.14 Schematic of the solutions on the r-F plane. The stable and unstable nullclines are indicated in blue and red respectively. 49

4.15 Schematic of the cochlear model bifurcation mechanism for several input signal amplitudes. 50

4.16 Adaptation of an oscillator to an external signal at different time steps. 52

4.17 Adaptation of an oscillator's frequency ω (blue line) to the frequency of an external signal ω_0 (orange line). 52

4.18 Training 4 oscillators on a signal composed out of 4 components with corresponding frequencies. 55

4.19 Training 4 oscillators on a signal composed out of 4 components with non-corresponding frequencies. 56

4.20 8 oscillators and 4 components, the 4 frequencies are captured by four oscillators while the four others stand quiet. 57

4.21 40 oscillators and 4 components, the 4 frequencies are clearly appearing on the image representing the different sine wave of the input signal. 58

4.22 100 oscillators capturing the information of an audio signal. 58

5.1 General overview of the thesis. 63

5.2 Evolution of the baseline implementation of SincNet: the grey graph shows the initial filter distribution and the blue graph shows the filter distribution after training. The x-axis represents the frequency range and the y-axis the amplitude of the filters. The filters themselves are represented by their central frequency (dot) and their bandwidth (bar). 67

5.3 Filter repartition of superimposed filterbanks before (top plot) and after (bottom plot) training. In the initialization, the red filterbank is a narrow-band filterbank composed of 30 filters. The rest are filterbanks of 10, 5 and 1 filters capturing information that could in principle be reconstructed by a combination of the narrow band filters. 71

6.1 General overview of the thesis. 75
6.2 A schematic overview of the original SincNet implementation model used as baseline in our previous work, the wav2vec2 fine-tuning path and the proposed fine-tuning path in this work. Based on compositionality capacity of networks, we combined the feature extractor of the original SincNet model with the pre-trained transformer of wav2vec2. 84
6.3 Training curve of the long-run training of 100 k updates. (**a**) depicts the WER of the validation set (dev-other), (**b**) contains the negative log likelihood of the loss, and (**c**,**d**) depict the learning rate evolution of the feature extractor and context network, respectively. 86
6.4 A more schematical explanation of the behavior of the network training explained in Figure 6.3. There are 2 training phases: one up to 40 k updates where the whole model except the feature extractor is frozen, and the second phase, where everything fine-tunes together. 87
6.5 Comparison of filter distribution for different run lengths and internal SincNet structure. Using maxpooling within the filter structure makes some wide-band filters appear in low frequency ranges. 88
6.6 Loss curve during **(a)** training and **(b)** validation. 90
6.7 Comparison of the content of the normalized filter sum using SincNet filters and using a same-size generic CNN. 91
6.8 Filter distribution after the self-supervised (ss) training and fine-tuning (ft) of the wav2vec2 model. Globally, filters do not tend to move significantly during fine-tuning when they have been incorporated during pre-training. 93
6.9 Distribution of the different filters in the frequency domain. The dots represent the central frequencies and the lines represent the bandwidths, similarly to Figure 5.2. The number of initial filters are (**a**) 100, (**b**) 80, (**c**) 60, and (**d**) 40. . . 94
6.10 Filter distribution after training for 40 filters. 96

7.1 General overview of the thesis. 99
7.2 Learning mechanism: division of different learning routines over the different parts of the model according to the physiological equivalent. 100
7.3 Hopf oscillator module schematic the oscillator output corresponds to the real value of the oscillator $r\cos\theta$. 103
7.4 Evolution of the r-value in the oscillator when $\mu = 1$ is fixed 105
7.5 Evolution of the r-value in the oscillator when μ is initialized to 0 and evolves over time for μ_{max} fixed to 0. 105

7.6 The evolution of the radius r of an oscillator for different signals tuned at its central frequency. 106
7.7 Cube root relationship between the input signal amplitude (A) and the oscillator radius (r). 106
7.8 Evolution of the radius of an oscillator radius for in input signal tuned at a frequency outside of the frequency bandwidth of the oscillator with a zoom showing the transient. 107
7.9 Linear relationship between the input signal amplitude (A) and the oscillator radius (r). 107
7.10 The evolution of the radius r of an oscillator for different signals tuned at its central frequency with an adaptable bifurcation parameter. 108
7.11 Diverging tendencies from the cube root relationship between the input signal amplitude (A) and the oscillator radius (r) with μ_{max} fixed to 0.1. 108
7.12 Comparison of the output of SincNet filters and oscillators for composite signals of multiple sinusoids. 108
7.13 Comparison of frequency separation between Hopf oscillators and SincNet filters.108
7.14 Comparison of the Hopf oscillator output wit mel-distributed SincNet filter output for a small utterance: ’eight’. 109
7.15 Comparison of output of linearly distributed filters with a DFT computation of a four digit utterance: ’one, zero, nine, two’, ... 110
7.16 Adaptation of μ parameter to a speech input in a noisy environment for a five digit utterance: ’two, six, five, five, five’. 111
7.17 Computation time needed for different values of the adaptation frequency going from 8kHz to 100Hz. 112
7.18 Illustration of the oscillator output for different adaptation frequencies for the utterance: ’nine, nine, zero’. 113
7.19 Integration of Hopf module in a simple ASR baseline. 114
7.20 PER for different adaptation frequencies from 100Hz to 16kHz and a comparison with fixed filters. 115
7.21 PER vs time consumption when introducing a for loop with CNN adaptation rates of 5Hz, 10Hz, 20Hz and 50Hz around the CNN. 117
7.22 Attention feedback loop 118
7.23 PER results when adding a large feedback loop. 119
7.24 Weight and bias of the linear combination module after training. 120
7.25 Attention feedback loop 120

List of Tables

2.1 CHiME4 dataset summary with the number of utterances per subset. The noisy datasets contain the four types of noisy environments: bus, café, street and pedestrian area. 21
2.2 Summary of LibriSpeech dataset 21

3.1 Baseline results: the results are computed for two different decoding methods: 'ctc' for ctc beam search and 'att' for attention rescoring. Five subsets are chosen: clean, real dev (RD), simu dev (SD), real eval (RE) and simu eval (SE) according to table 2.1. 29
3.2 Results of fixed routing with attention rescoring decoding method. 29
3.3 Number of utterances in each category for 2 experts 30
3.4 Number of utterances in each category for 3 experts 30
3.5 Number of utterances in each category for 5 experts 31
3.6 Results of learned routing mechanism 32

4.1 Phone error rate comparison between MFCC and CARFAC features. 60

5.1 Filter pruning experiment: numbers of narrow band filters and related PER in function of the initialization. 68
5.2 SincNet experiment: compare the performance of the training with the filters fixed and the filters that are free to train. 68
5.3 Mean Euclidean distance between narrow bandfilter's normalized central frequency distribution and different scalings for different amount of filters (Mel filterbank) and different initial scalings (30 filters). 69
5.4 Summary of experiments using narrow and/or wide-band filters 71
5.5 The effects of modifying the downsampling and pooling schemes. The numbers in the second column refer to the downsampling rate at each of the pooling operations in the convolutional layers (1 implies no downsampling). 73

6.1 Comparison between a short and long run using no maxpooling after the filter layer in the first 2 experiments and with a maxpooling that has a kernelsize of 3 in the third experiment. Thus, the third experiment has a downsampling of a factor 3 just after the sinc filters via a maxpooling operation in the time domain. 87

6.2 Table summarizing the time of each experiment on 4 parallel GPUs and the number of parameters contained in the feature extractor. Note that this number of parameters is quite small compared to the rest of the network (90 M parameters). 89
6.3 Performance of the different experiments on the capacity of SincNet (SN) to replace the initial CNN structure with a 95% confidence interval, assuming the result is beta distributed. 90
6.4 Comparison of the loss and accuracy of the baseline pre-trained model and the model after a further pre-training for both frozen and trainable filters. After 10 k updates, the model already performs well but not as good as the baseline, while after 100 k, the model reaches the performance of the baseline in terms of loss and accuracy. 92
6.5 Performance of model using trainable filters in pre-training (100 k updates) and fine-tuning (10 k updates). 92
6.6 Performance on the dev-clean subset of LibriSpeech for different numbers of filters. 95

7.1 Input and output size of the CNN for different CNN adaptation rates. 117
7.2 PER for the different SNR levels on the TIMIT dataset with a model trained on clean speech. 122
7.3 PER for the different SNR levels on the TIMIT dataset with a model trained on noisy speech 123

Glossary

AGC Automatic Gain Control.

ASR Automatic Speech Processing.

CARFAC Cascade of Asymmetric Resonators with Fast-Acting Compression.

CN Cochlear Nucleus.

CNN Convolutional neural network.

CP Critical Point.

DFT Discrete Fourier Transform.

DNN Deep Neural Network.

E2E End-to-end.

fMRI functional Magnetic Resonance Imaging.

GMM Gaussian Mixture Model.

GPU Graphics Processing Unit.

HMM Hidden Markov Model.

IC Inferior Colliculus.

IHC Inner Hair Cell.

LLM Large Language model.

LOC Lateral Olivary complex.

MFCC Mel-Frequency Cepstral Coefficients.

MGB Medial Geniculate Body.

ML Machine Learning.

MLP Multilayer Perceptron.

MOC Medial Olivary Complex.

NLP Natural Language Processing.

OAE Otoacoustic Emissions.

OC Olivary Complex.

OHC Outer Hair Cell.

PER Phone Error Rate.

PLL Phase Locked Loop.

PLP Perceptual Linear Prediction.

RMS Root Mean Square.

SNR Signal-to-Noise Ratio.

SOC Superior Olivary Complex.

TCNN Transposed Convolutional neural network.

WER Word Error Rate.

WSJ Wall Street Journal.

1 Introduction

Cochlear models have been a subject of research for a while. The understanding of the human ear is of great interest both in the neuroscience field for a better understanding of the underlying neurological processing and in the automatic speech recognition (ASR) field as a source of inspiration for technology. However, research in those two fields has evolved separately, linked to the differences in their objectives: neuroscience seeks to better understand the neurological system for medical applications, while ASR research focuses on achieving the best accuracy and efficiency. This thesis brings together the latest foundation models for ASR with the latest understanding of the cochlea, which involves a gain adaptation mechanism and the efferent pathway.

1.1 Context and Motivation

This thesis lies at the intersection of two fields: the physiological understanding of the cochlea and auditory system, and ASR. The cochlea, an organ in the inner ear, transforms sound waveforms into neural signals (Von Békésy, 1960; Webster, 1966). ASR, a technology within computer science, converts spoken language into written text (Amari, 1993; Baevski, Zhou, et al., 2020; Collobert et al., 2016; Morgan & Bourlard, 1990; Rabiner, 1989).

Although these fields differ in computational objectives, they share a common underlying process: transforming human speech into comprehensible signals, be they neural signals, phonemes, or words. In studies of the human auditory system, researchers aim to understand the workings of the cochlea, the auditory pathway to the midbrain, and the auditory cortex. This entire pipeline helps scientists model how the nervous system processes speech signals and converts them into neural representations that the brain can interpret. On the ASR side, the primary objective is to develop systems capable of converting spoken language into phonemes, words, or other lexical units and that are computationally efficient.

ASR technology initially drew significant inspiration from models of the human auditory system (Von Békésy, 1960). However, as computational resources expanded and model performance improved, research in these two fields began to diverge. Despite this separation, key advances have often emerged when the fields draw inspiration from each other. For example, ASR has evolved to process raw speech input rather than relying solely on precomputed features (Palaz et al., 2013a). Meanwhile, in auditory neuroscience, researchers have discovered correlations between auditory brain signals and outputs from pre-trained ASR systems. This thesis combines physiological modeling of the cochlea with the use of ASR to improve the *explainability* of certain physiological behaviours.

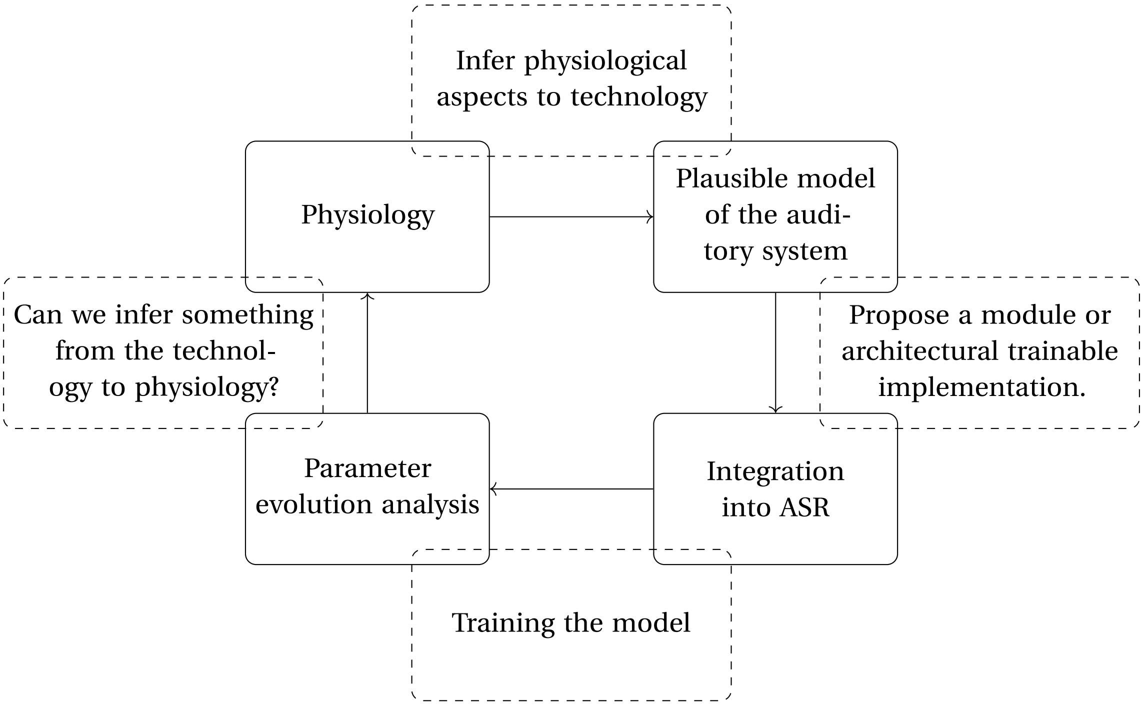

Infer physiological aspects to technology
Physiology
Plausible model of the auditory system
Can we infer something from the technology to physiology?
Propose a module or architectural trainable implementation.
Parameter evolution analysis
Integration into ASR
Training the model

Figure 1.1: Overview of the research strategy. Inspired by the human auditory system, this study explores a plausible cochlear model integrated into a neural network, aiming to provide insights into both hearing technology and human physiology.

As illustrated in Figure 1.1, advancements in hearing technology have been inspired by physiological models. These range from simple logarithmic frequency mappings to more complex representations of cochlear mechanics (Gold, 1948; Kemp, 2002; Von Békésy, 1960). Building on this knowledge and the performance of existing technologies, this work focuses on the development of physiologically plausible models.

On the one hand, a basic understanding of the cochlea models it as a filterbank (Ravanelli & Bengio, 2018a; Zeghidour et al., 2018). On the other hand, a more detailed understanding involves modelling the organ of Corti as a Hopf oscillator, operating near the Hopf bifurcation (Camalet et al., 2000; Hudspeth et al., 2010b). This mathematical framework has been shown to accurately capture key aspects of the cochlear function.

1.2 Problem statement, research objectives and limitations

On the one hand, our understanding of the human cochlea has evolved significantly over the years. Initially conceived as a simple filterbank (Von Békésy, 1960), the cochlea's inner workings are now recognized as far more intricate (Brownell et al., 1985; Gold, 1948). Moreover, interactions along the auditory pathway have revealed feedback loops that contribute to the finer calibration of signals.

On the other hand, ASR models have also undergone substantial development. From early systems composed of manually designed statistical units (Morgan & Bourlard, 1990; Rabiner, 1989), the advent of deep learning has introduced trainable models of increasing complexity and scale (Baevski, Zhou, et al., 2020; Baevski et al., 2022; Collobert et al., 2016; Hinton et al., 2012). While these large models have achieved remarkable improvements in performance, they have also reduced interpretability and limited comparability to the human auditory system.

Bridging the gap between these domains offers intriguing possibilities for improving interpretability but comes with trade-offs, such as reduced performance and computational inefficiency due to the complexity of such implementations. Addressing these challenges is a central focus of this thesis. Additionally, advancing ASR research often involves competing against well-resourced teams with access to extensive computational capacities, making it unrealistic to aim for state-of-the-art performance under limited resources.

The primary goal of this thesis is, therefore, not to surpass the latest ASR models in terms of efficiency or performance but to propose novel approaches and insights. These approaches draw inspiration from the physiological workings of the human cochlea. This is achieved by integrating a deeper understanding of cochlear mechanisms into the ASR domain. Another key objective is to analyze performance changes in various experiments to better understand the learning processes of the human auditory system.

Initially, this thesis sought to combine state-of-the-art ASR models, such as pretrained transformer architectures, with a mathematical model of the latest understanding of the cochlea. However, the inherent complexity of both systems made such integration computationally prohibitive, requiring resources far beyond reasonable limits. Nevertheless, this work explores both fields to a feasible extent. Specifically, it investigates:

1. The integration of trainable filters within a state-of-the-art self-supervised model context.

2. The incorporation of active amplification oscillators into a simpler ASR framework.

The pathways and building blocks leading to these objectives are further detailed in Section 1.3.

1.3 Outline and main contributions

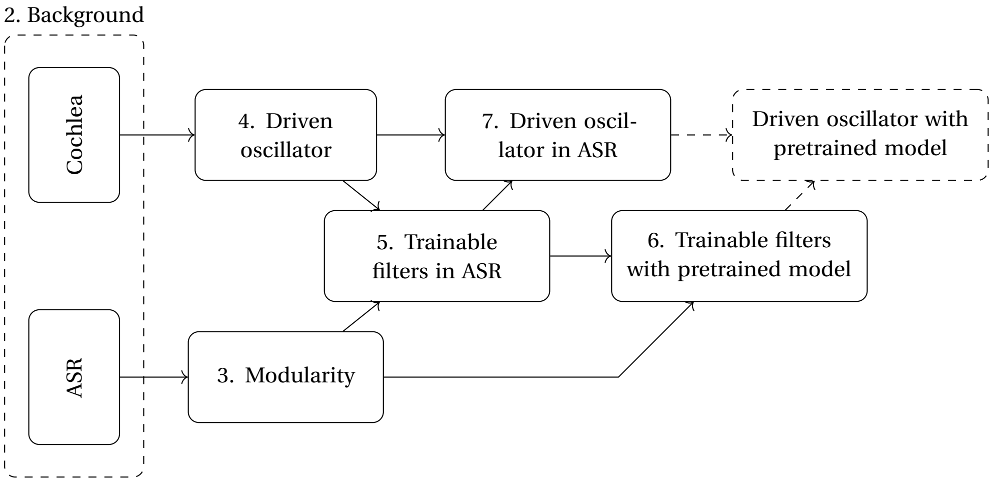

2. Background
Cochlea
ASR
4. Driven oscillator
7. Driven oscillator in ASR
Driven oscillator with pretrained model
5. Trainable filters in ASR
6. Trainable filters with pretrained model
3. Modularity

Figure 1.2: General overview of the thesis.

This thesis is built on two main pillars: ASR and the physiology of the cochlea. To provide a clear understanding of the thesis structure, Figure 1.2 summarizes the overall framework and illustrates the interconnections between the different chapters.

- Chapter 2 provides background information, presenting the state-of-the-art understanding of cochlear function, existing cochlear models with varying levels of physiological granularity, and the key advancements made in the field over the past decade. The chapter highlights the benefits and inherent limitations of these developments.

- Chapter 3 introduces the concept of modularity (Pfeiffer et al., 2023) and explores how modularity can serve as a bridge between cochlear models and ASR systems. It also presents an investigation into modularity (Pfeiffer et al., 2023) within a conformer-based ASR system. Specifically, we demonstrate that incorporating modularity enhances system performance. Further analysis explores the deployment of both fixed and learned routing mechanisms, applied to a noisy dataset containing speech in various noisy environments. Employing modularity to distinguish between noise types and clean speech improves the learning curve and overall performance across all dataset subsets. This investigation laid the groundwork for modularity to play a central role in subsequent chapters, bridging the gap between pure ASR systems and physiological cochlear models.

- Chapter 4 delves into the active amplification mechanisms of the cochlea, demonstrating how Hopf oscillators effectively mimic these mechanisms through bifurcation dynamics. This chapter acts as a prior work chunk to demonstrate the effectiveness of Hopf oscillators to encode speech before integrating an active amplification oscillator based system into an ASR model in chapter 7.

- Building on the modularity concept of chapter 3, chapter 5 investigates the integration of trainable filters into ASR systems. Using SincNet (Ravanelli & Bengio, 2018a, 2018c), as a foundation, we analyze the characteristics of trainable filters when they adapt within a small ASR context. The analysis reveals that the filters tend to form a filterbank of 30–40 filters, with some filters capturing wideband information. This behaviour aligns with neurophysiological observations: at the cochlear level, narrowband filters exhibit tonotopic organization, capturing frequency-specific information, while higher levels in the auditory pathway combine neural information in a wideband manner.

- Chapter 6 further builds on the work in chapter 5, extending the concept of trainable filters to larger ASR systems based on self-supervised pretrained models. Modern state-of-the-art ASR systems leverage models pretrained on large datasets, which are subsequently fine-tuned on smaller datasets to achieve superior performance compared to traditional ASR methods. Extending the modularity concept, we combine a trainable filterbank with a pretrained transformer model. This study reveals that, within a larger pretraining context, wideband filters no longer emerge. This suggests that in transformer-based pretrained models, broader combinations of narrowband filters are constructed at higher layers during pretraining. Notably, the model consistently establishes a baseline of approximately 40 filters to capture activity across different frequency ranges.

- Chapter 7 combines the latest understanding of the cochlea as a Hopf oscillator-based model poised at the Hopf bifurcation with the ASR baseline used in chapter 5. We propose an implementation using mel-spaced oscillators covering the speech frequency range. These oscillators exhibit cube-root compression and operate at the Hopf bifurcation. We demonstrate the application of this module in an ASR system and propose the addition of a larger feedback loop, inspired by physiological evidence of an efferent path involving higher-order nuclei in the auditory pathway. Incorporating this larger feedback results in a significant performance improvement. The performance of this approach is evaluated on both clean and noisy datasets.

- The initial objective of this thesis was to integrate a complex cochlear model with a pretrained ASR system. However, due to computational and time constraints, such experiments were deemed infeasible.

- Chapter 8 concludes the thesis by summarizing the key findings and offering recommendations for future research.

In order to improve readability, much of the prose in this thesis has been refined by a large language model (LLM) explicitly prompted to correct grammar [I].

[I] A local instance of Llama 3.2 3B prompted with "Can you correct the following text:"

2 Background

In this chapter, we cover the background topics required to understand the general aspects of the thesis. The thesis is constructed on two main pillars. On the one hand, we have the understanding of the physiological mechanism of the cochlea and the existing cochlear models; and on the other hand, the working of ASR. Figure 2.1 highlights the background as foundations on which the rest of the thesis is built. The understanding of the cochlea lies at the foundation of the cochlear model we want to build. In contrast the ASR system serves as the container in which the performance of those models can be tested.

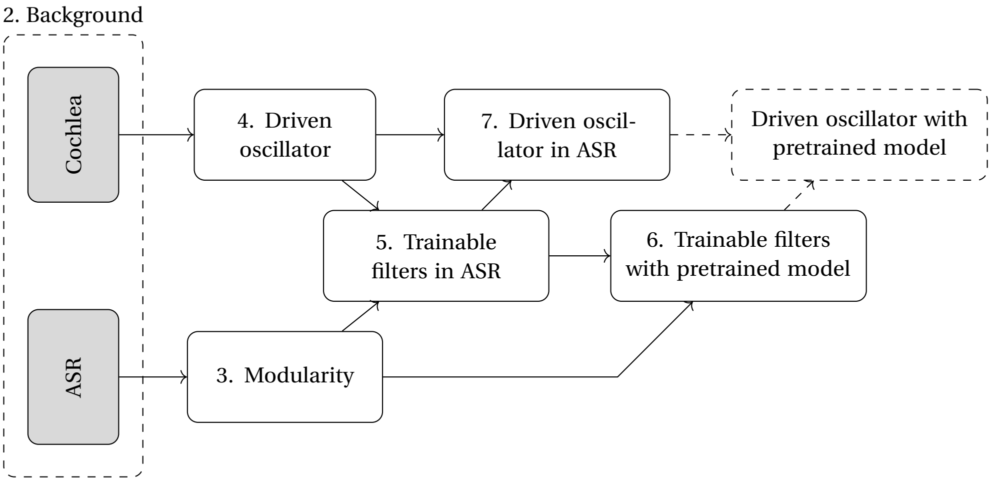

2. Background
Cochlea
ASR
4. Driven oscillator
7. Driven oscillator in ASR
Driven oscillator with pretrained model
5. Trainable filters in ASR
6. Trainable filters with pretrained model
3. Modularity

Figure 2.1: General overview of the thesis highlighting the background section.

2.1 The cochlea

Research in the neurophysiological field has investigated the intricate relationship between the cochlea and the brain. This section begins with a broad overview of the general functioning of the cochlea as understood since the early 20th century. Next, a second subsection delves into the working of the organ of Corti and the active amplification mechanisms that occur within it.

The organ of Corti's role in generating oscillations is further supported by an active feedback loop linked to the hair cells. Additionally, research on the auditory path in the midbrain suggests that larger feedback loops play a significant role in the hearing process. The presence of some larger feedback loops as well as the reason of their utility is still a research question in the auditory field. An overview of the main feedback loops is presented in subsection 2.1.4 .

2.1.1 Anatomy and tonotopic organisation

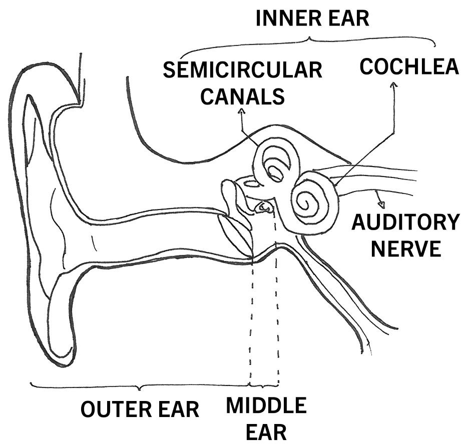


Figure 2.2: Schematic of the human ear divided into outer, middle and inner ear.

To gain insight into the workings of the cochlea, this section begins with a concise review of its anatomical structure (Webster, 1966). The human ear is composed of three distinct parts: the outer ear, middle ear, and inner ear. Figure 2.2 illustrates the overall organization of these components. The outer and middle ears function as low-pass filters that amplify sound, while the inner ear is comprised of two main structures: the cochlea and semi-circular

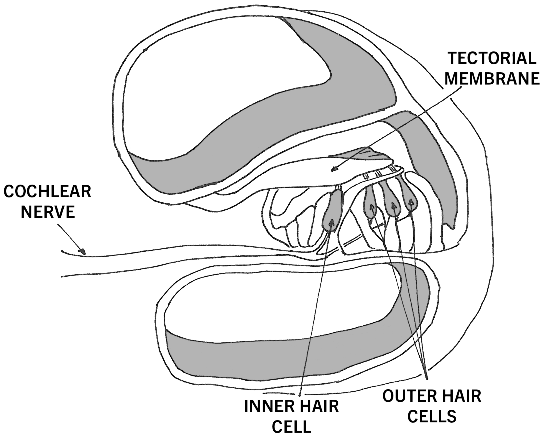


Figure 2.3: Schematic of the organ of Corti

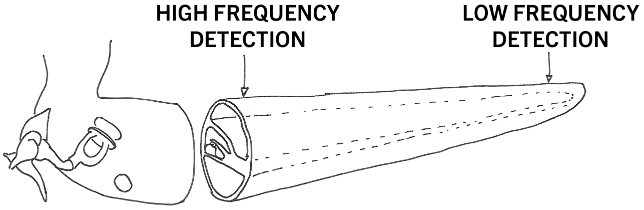


Figure 2.4: Schematic of the cochlear duct, the central membrane is an approximation of the frequency-selective basilar membrane. The high frequencies are detected on the beginning of the cochlear duct and the lower frequencies are detected at the end, near the apex.

canals. The cochlea, located within the inner ear, serves a critical role in hearing, whereas the semi-circular canals contribute to balance and equilibrium. The cochlea itself consists of three distinct channels, defined by the Reissner membrane and Basilar membrane. These channels are divided into two outer and one inner channel.

The organ of Corti (as shown in Figure 2.3), is a complex sensory apparatus located on the basilar membrane within the inner channel of the cochlea. The organ of Corti consists of three primary components: the tectorial membrane, inner hair cells (IHC), and outer hair cells (OHC). In addition to its structural complexity, the organ of Corti is also innervated by the cochlear nerve. This nerve serves as a critical conduit for the transmission of neural information between the cochlea and the rest of the auditory pathway.

2.1.2 High-level working of the cochlea

Acoustic waveforms initially reach the outer ear, where they are transmitted through the eardrum to the middle ear. The ossicles of the middle ear match the impedance between the ear drum and the oval window, ultimately causing the acoustic waves to travel to the inner ear. At this point, a transmission to perilymph occurs. The perilymph is a fluid situated in the outer channels of the cochlea. This liquid as well as the protein structures and their properties around the cochlea has a filtering function: it makes the vibrations progress through the cochlea according to their frequency, the lower the frequencies, the further the waves are propagated. As a result, the basilar membrane vibrates at specific frequencies that correspond to particular locations along its length. These vibrations can be detected by the interactions within the Organ of Corti, allowing for the processing and analysis of sound.

This high-level understanding of the cochlea, as proposed by Von Békésy (1960), can be approximated using a series of filters: higher frequency signals are detected earlier in the cochlear duct, while lower frequency signals are detected later near the apex (as shown in Figure 2.4). Filterbanks have been widely employed to define typical feature computations and build non-complex models for ASR tasks. However, the earliest studies were limited because they were conducted on the cochleae of deceased animals, which lack neural feedback, resulting in passive filterbank behaviours. Subsequent studies on the cochlea revealed that its actual operation involves an active amplification mechanism (Gold, 1948; Kemp, 1978; Zurek, 1981).

2.1.3 The active amplification mechanism

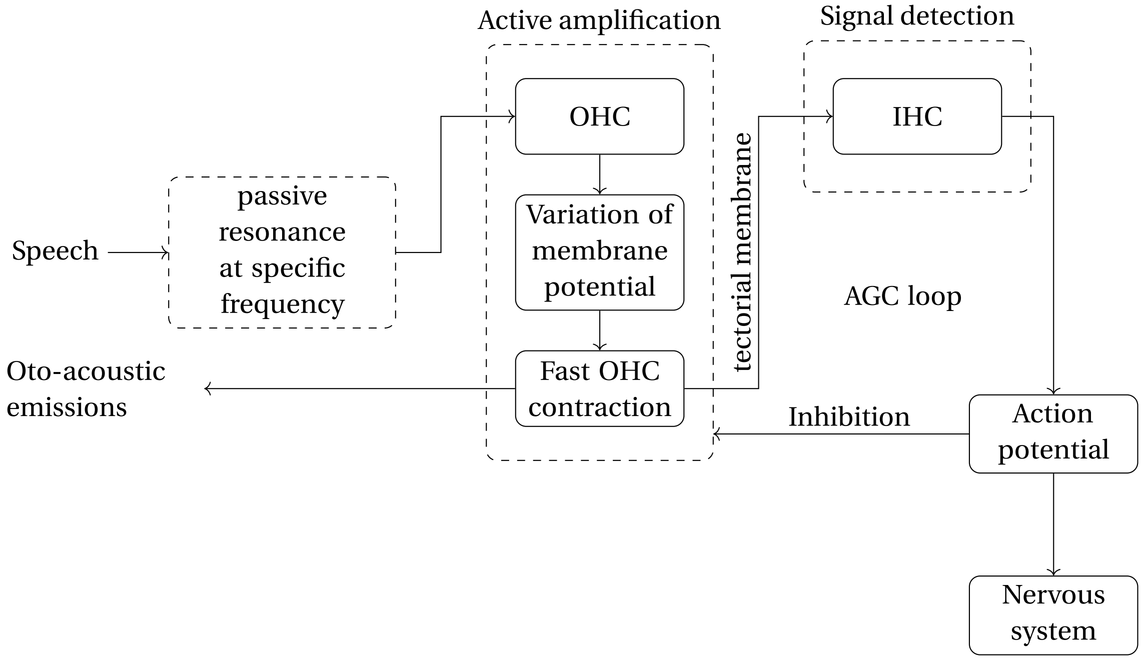


Figure 2.5: Schematic of the working of the cochlea

Further research on the organ of Corti by studying living animals revealed a more detailed understanding of the cochlea. Contrary to initial assumptions that it behaves like a passive filterbank, studies in the 1980s showed that the cochlea is an *active amplification oscillator*. Gold (1948) had the intuition that a passive oscillation of the basilar membrane could not be enough to detect sound. He then suggested that an active amplification mechanism of the acoustic signals was present in the cochlea. This hypothesis was later confirmed by the phenomenon of otoacoustic emissions (OAE). OAEs are vibrations produced by the inner ear in response to auditory stimuli or even spontaneously, without external stimulation. Measured in human ears in the late 1970s and early 1980s (Kemp, 1978; Zurek, 1981), OAEs provided evidence for the existence of nonlinear mechanisms within the cochlea that responded mechanically to acoustic stimulations. Further studies revealed that the OHCs possess a unique electromotile capacity (Brownell et al., 1985), which is responsible for the production of OAEs. (Geisler, 1986; Neely, 1993) The IHCs are the real sensory receptors that connect the cochlea to the nervous system through electric pulses (Russell & Sellick, 1977). Figure 2.5 illustrates the connections between these different elements. The signal is first filtered out by the perilymph at different frequencies between 20Hz and 20kHz, depending on the location on the cochlear duct. This induces a vibration of the basilar membrane at different frequencies with a passive resonance. The OHCs detect these vibrations, leading to changes in their membrane potential and subsequent fast contractions. The input vibration is actively amplified by the OHCs, generating larger vibrations of the tectorial membrane that can then be detected by the IHC. This active amplification mechanism is then inhibited by feedback signals, allowing for more efficient sound detection. Finally, the signal is transmitted to the brain.

2.1.4 Feedback connections in auditory path.

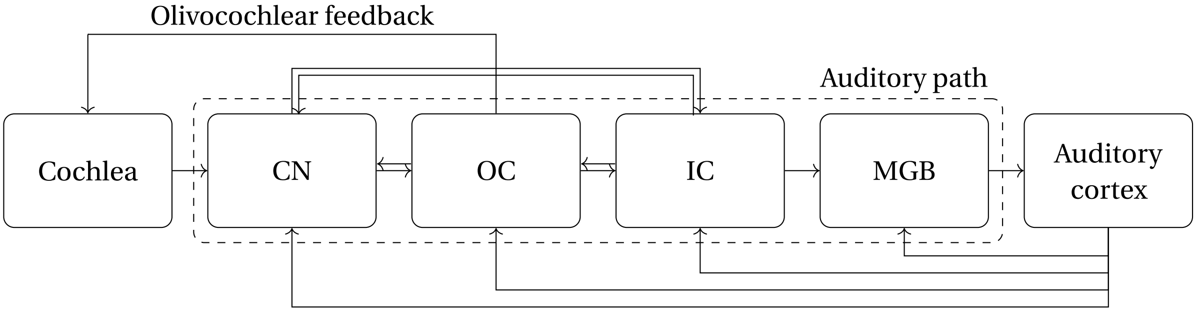


Figure 2.6: Auditory path from cochlea to midbrain with the efferent paths from the auditory cortex and mibrain.

The auditory pathway involves a complex interplay of neurons from the cochlea to the auditory brain, with multiple feedback mechanisms that refine and optimize sound processing. While the efferent path in the auditory brain is well established, the various feedback loops that comprise it are multifaceted and serve distinct purposes. Those feedback loops are schematized in Figure 2.6.

The olivocochlear feedback loop is a well-known mechanism connecting the olivary complex (OC) with the cochlea across different mammals (Rasmussen, 1946). This efferent pathway has been favoured by evolutionary selection, as evidenced by its widespread presence in various species (Romero & Trussell, 2022). The olivocochlear feedback loop can be further divided into two main branches:

1. The medial olivary complex (MOC) to the OHC, which serves to protect against damage from loud noise and improve speech perception in noisy environments (D. W. Smith & Keil, 2015).
2. The lateral olivary complex (LOC) to the synapses of the IHC, modulating auditory nerve sensitivity (Warr et al., 1997).

Additionally, the olivocochlear feedback loop also influences the excitability of the cochlear nerve, which is essential for slowing cochlear aging (Liberman et al., 2014; Maison et al., 2013).

The descending projections from the auditory cortex form two distinct loops: the colliculo-thalamic-cortico-collicular loop and the bottom loop (Terreros & Delano, 2015).

- The top loop involves descending projections to the medial geniculate body (MGB) and inferior colliculus (IC), forming a tonotopic feedback loop that modulates sound intensity, frequency, and spatial processing.
- The bottom loop comprises descending projections to the cochlear nucleus (CN) and superior olivary complex (SOC), influencing cochlear responses.

Furthermore, IC, SOC, and CN are interconnected within a bidirectional network that facilitates top-down modulation in the auditory pathway (Terreros & Delano, 2015).

2.2 Broad overview of cochlear Models

In both the neurophysiological and technological fields, researchers have developed cochlear models to gain insights into their functioning and generate plausible outputs.

In the neurophysiological field, the primary goal of these models is to elucidate the intricacies of the cochlea itself. As such, these models are often more computationally intensive and strive to capture the physiological reality with greater accuracy. By doing so, researchers can better understand the complex mechanisms underlying sound processing in the ear.

In contrast, applications of cochlear models in the ASR field have two distinct uses:

1. A separate feature extraction : These models can be used to generate features that are computationally intensive and detailed. This approach allows for the creation of high-fidelity representations of speech sounds, which can be particularly useful in challenging acoustic environments and afterwards be used to train models.
2. Incorporation into ASR systems : However, when it comes to integrating cochlear models into ASR systems, computational compromises are often necessary, but parameter training of cochlear models is made possible.

By striking a balance between computational intensity and practicality, researchers can develop cochlear models that not only advance our understanding of sound processing but also enable more effective applications in ASR.

Existing cochlear models typically concentrate on a particular aspect of the cochlea's functioning and can be implemented based on analyses conducted at various scales. At the macro scale, filterbank models attempt to simulate the cochlea's output signals by adapting filters, such as Gammatone filters. These models aim to capture the overall structure and functionality of the cochlea, with a focus on its acoustic properties. In contrast, models that delve deeper into the micro scale focus on specific components of the cochlea, such as: the molecular interactions at the neurotransmitter level, the fluid dynamics coupled with the basilar membrane motions and the OHC-IHC interaction. By targeting specific aspects of the cochlea's functioning at different scales, researchers can develop more nuanced and accurate models that capture the intricacies of sound processing in the cochlea.

2.2.1 Gammatone filter

Many ASR features are based on filterbanks. Among these, the Gammatone filter is particularly well-suited to mimic the output of the cochlea. The Gammatone filter (Johannesma, 1972) is a type of filter that provides a good approximation of the impulse response shape of the auditory system. This filter combines a gamma-distribution envelope with a polynomial increase and exponential decrease, along with a cosine function at a specific frequency. The mathematical representation of the Gammatone filter is given by Equation 2.1:

$$gt(t) \propto t^{n-1} \exp(-2\pi bt) \cos(2\pi f_0 t + \phi) \tag{2.1}$$

The Gammatone filter has been used as a tool to model the cochlear response to a stimulus (Russo et al., 2019). One of its key benefits is that it can approximate both the cochlear fluid filtering and the OHC amplification. In a noisy environment, Gammatones tend to perform better than the standard ASR features such as MFCCs (see section 2.3.3). Researchers have also explored using Gammatone filterbanks for cochlear implants, with proposals made by Karuppuswamy and Arumugam (2013), Ngamkham et al. (2010), and Tabibi et al. (2017).

2.2.2 Neurotransmitter based and spiking models

The physiological function of the cochlea is to transform mechanical vibrations into neural activity pattern, involving transfer of neurotransmitters to emit a spike. A model based on the computation on the flow of neurotransmitters through synapses is the Meddis Hair Cell Model (Meddis, 1986), which focuses on the transduction process in the IHC to convert mechanical vibrations into neural signals. This model operates at a synaptic level, calculating the amount of transferred neurotransmitters and determining firing probabilities in the auditory nerve. This approach is particularly useful for modeling auditory nerve responses or integrating it into larger auditory systems. In contrast, X. Zhang et al. (2001) proposed an auditory nerve model that focuses on the transduction of sound vibrations into spikes, combined with a global understanding of the cochlea to simulate the hearing function. This model is well-established in the neuroscience domain. Additionally, Cramer et al. (2020) proposed a spike-dataset production model to utilize spike features as input for speech technologies and (Yang et al., 2016) proposed a hardware implementation that further has been combined with deep learning models for ASR (Liu et al., 2013; Wang et al., 2022).

2.2.3 Basilar membrane coupled to fluid dynamics

Hydrodynamic models of the cochlea primarily focus on the sound propagation through the perilymph (cochlear fluid). Vibrations are introduced into the cochlear fluid through the ossicles in the middle ear and the oval window, triggering fluid movements that cause the basilar membrane to vibrate at frequency-dependent locations.

There are several notable examples of hydrodynamic models:

- Allen's Cochlear Model (Allen, 1980) is characterised by a tonotopic organization of the cochlea, fluid dynamic equations and a representation of the basilar membrane as an elastic structure with varying stiffness and damping along the frequency axis.

- Steele and Taber Models (Steele & Taber, 1979) employ the Wentzel-Kramers-Brillouin (WKB) asymptotic method to simulate the fluid dynamics within the cochlea. These models are compared to finite difference calculations to validate their accuracy.

- Zweig Cochlear Model (Zweig et al., 1976) focuses on nonlinear basilar membrane responses in the cochlear tuning.

2.2.4 Hair cell interaction mechanism

Some models focus primarily on the interactions between the OHC and the IHC, as well as the inhibitory feedback loop between these elements. One notable example is cascade of asymmetric resonators with fast-acting compression (CARFAC) (Lyon, 2017a), a model that simulates these interaction using an active gain control (AGC) loop. This mechanism amplifies sound at specific frequencies, mimicking the behaviour of the cochlea. Another approach uses Hopf oscillators poised at the Hopf bifurcation. This mechanism reflects mathematically the working of the cochlea.

The work presented in this thesis builds upon and further explores these interactions, focusing on the intricate mechanisms that govern the OHC-IHC interaction and the inhibitory feedback loop.

CARFAC

The CARFAC model is a comprehensive model that incorporates various physiological behaviours characteristic of the ear. A general overview of the CARFAC architecture is depicted in Figure 2.7, providing a visual representation of the interactions between the different components. The construction of the CARFAC model is inspired by the physiological workings of the cochlea, as presented in section 2.1.3. The model proposes an implementation that closely approximates the output and function of each physiological element. The frequency filtering by the non-uniform liquid medium is modelled using second-order low-pass filters H with undamped resonance frequencies. These filter frequencies are distributed according to the Greenwood distribution (Sridhar D, 2006). The transfer function of the second-order filters

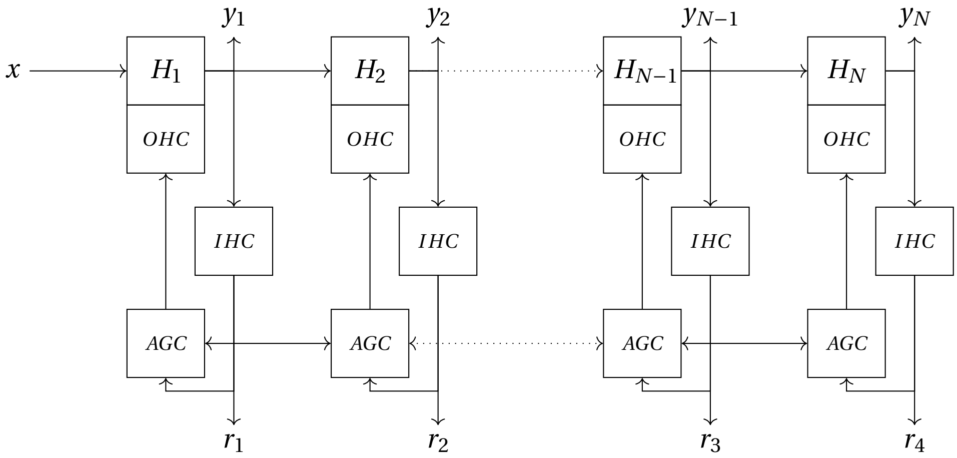


Figure 2.7: Schematic of CARFAC from Lyon (2017a). The y-components represent the state of the basilar membrane and the r-components represent the signals sent to the brain.

used in CARFAC is defined by the following equation:

$$H(z) = \frac{z^2 + (-2a_0 + c_0 h) r z + r^2}{z^2 - 2a_0 r z + r^2} \tag{2.2}$$

The denominator of this equation defines the poles which determine the resonance frequencies. The numerator defines the zeros, which are coupled to the poles and cause a steeper response on the high frequency side. This asymmetry gives control over filter shape and gives a flat high-frequency asymptote. The CARFAC cascade is organized in a manner similar to that of the cochlea, with the highest frequencies first due to the cascaded structure. The output of each filter is progressively transferred to the following resonator, which amplifies the signal at its resonance frequency and filters out higher frequencies. In CARFAC, OHCs can actively amplify the signal at the resonance frequency of the corresponding resonator. This is achieved by modifying the radius of the zeros and poles of the second order filters in the z-plane, which affects the damping of the resonators and the amplitude of the output signal y, representing the basilar membrane vibration. The movement of the basilar membrane is detected by the IHCs, which convert these signals into action potentials (neural activity patterns). The IHC possess the unique property of acting as half-wave rectifiers, transforming vibrations into spikes. Further, an AGC model fine-tunes the OHC undamping mechanism to adaptively adjust based on the output of the IHC. The AGC blocks are connected throughout the cascade of asymmetric resonators.

In the electronics field, a stable and scalable implementation of the CARFAC model has been realized on an Field Programmable Gate Array (FPGA) by Xu et al. (2018) for cochlear modelling purposes.

Hopf oscillator model

The Hopf oscillator model, originaly proposed by Hopf (1942) has been further developed and adapted as a suitable model for cochlear applications by Martin and Hudspeth (1999). This model employs a mathematical approach that focuses on capturing the intricate inner behaviour of the interactions, rather than merely simulating the output. The two key features of this model are the cube root compression and its ability to exhibit a Hopf bifurcation, which captures the active amplification and damping mechanisms. The complexity of this model makes it an attractive choice for modeling cochlear dynamics in this thesis. The mathematical behaviour will be further detailed in Chapter 4.

Furthermore, an electronic implementation a Hopf reservoir, under the form of an electronic circuit has been proposed for sound recognition (Shougat et al., 2021, 2023), which shows interesting results for hardware implementations.

2.3 ASR

2.3.1 Evolution of ASR

The first ASR system to Hidden Markov Models (HMMs)

ASR began in the mid-20th century with analog devices capable of recognizing a limited set of words. One of the first prototypes, called Audrey, was developed by Bell Labs and focused on automatic digit recognition. As research progressed, ASR systems started handling larger vocabularies through techniques like template matching. In the 1980s, a stochastic approach gained popularity, with HMMs becoming widely adopted for ASR (Rabiner, 1989). HMMs use states to represent entities such as phonemes, with probabilistic transitions allowing for movement between nodes or staying in the same node. An advancement in HMMs was the integration of Gaussian Mixture Models (GMMs) into the nodes, enhancing robustness and enabling the models to better capture complex distributions in acoustic features. This approach required predefined features to be computed for HMM nodes, based on known speech features (Rabiner, 1989). Alongside GMM-HMM models, the combination of multilayer perceptrons (MLPs) with HMMs also demonstrated strong performance in ASR (Morgan & Bourlard, 1990).

Gradient descent mechanism

Parallel to the ASR first steps, neural networks began in the mid-20th century with the first MLP. This consisted of layers of neurons computing linear combinations of input units and followed by non-linear activation functions. Combined with a gradient descent mechanism, those first networks were able to be trained. The gradient descent mechanism keeps track of the path through which the signal has progressed when a batch of examples has been forwarded through the system, the difference between the real and predicted targetsknown as the loss. This loss is then backpropagated through the model and at every node, the gradient with respect to the obtained loss is computed. The gradient corresponds to the first order derivative of a given node that gives the direction towards which the value of the node should evolve. A learning rate then defines the size of the step in the gradient direction. By repeating this operation several times, the model learns node values that predict the right targets. Several loss computation mechanisms and different learning rate schedulers exist, allowing for adjustments to be made to the learning rate and optimization of the subsequent loss curve according to the specific model and task requirements.

Deep Learning models

Around 2010, several studies demonstrated that the performance of neural networks outperformed that of HMM based models (G. E. Dahl et al., 2011; Pan et al., 2012; Seide et al., 2011a). With the advent of deep neural networks, precomputed nodes are no longer necessary; instead, stochastic gradient descent allows the system to learn the essential characteristics needed for ASR through training (Amari, 1993). Initially, models used precomputed features such as Mel-frequency cepstral coefficients (MFCCs) as input. However, later research proposed end-to-end approaches that enabled the system to define its own features (Hinton et al., 2012; Palaz et al., 2013b). To train these models, large datasets are required, typically consisting of raw speech or preprocessed speech features accompanied by their corresponding transcriptions. The training process typically relies on an encoder-decoder structure that learns to recognize individual units of language, such as letters, phonemes, words, or other linguistic elements from raw speech or speech features. This approach is known as supervised learning, where every input sample has a corresponding target transcription.

Self-supervised models

Self-supervised learning was introduced by Collobert and Weston (2008) in natural language processing (NLP), and allows models to learn without labeled data. Unlike supervised learning, which requires labeled data and can be time-consuming to assign labels, self-supervised learning uses large amounts of unlabeled data to train the model.

Self-supervised models typically follow a two-stage training procedure:

- Pre-training: This is a resource-intensive process that creates a large pre-trained model creating orthogonal representations in a latent space. The training is based on a similarity measure between signals.
- Fine-tuning: The pre-trained model is then adapted to a specific task using smaller amounts of labeled or unlabeled data.

Self-supervised learning became particularly popular for transformer-based models. Transformers have shown to outperform classical deep learning modules such as recurrent neural networks, MLP and convolutional neural networks (CNNs) in a number of tasks such as NLP and increasingly in ASR and vision. Its mechanism is based on attention, which captures through keys, queries and values the structure of the different types of input (Vaswani et al., 2017). This structure is more robust, but is also more computationally demanding.

2.3.2 ASR structure

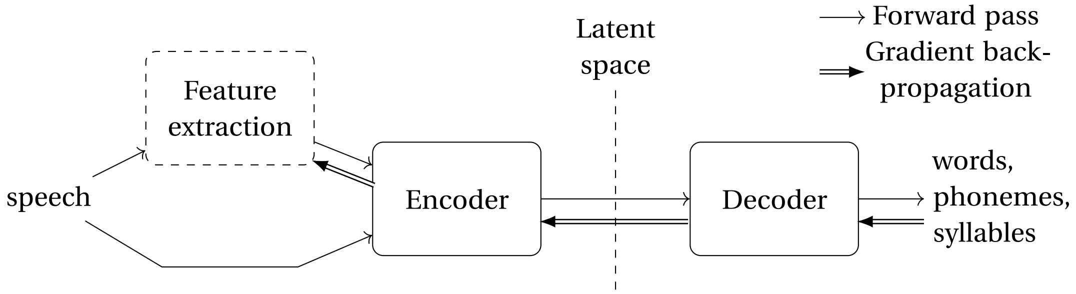


Figure 2.8: Structure of ASR system.

An ASR system typically consists of two main components: a speech or feature encoder and text decoder as illustrated in Figure 2.8. The encoder takes speech or speech features as input and maps them to a lower-dimensional latent space. The latent-space is a N-dimensional space in which speech features corresponding to the same target are grouped together. The decoder links the latent space representations to the different targets.

In self-supervised model, only the encoder is trained during the pretraining phase. This pretraining creates orthogonal representations for different inputs based on a predictive algorithm. During finetuning the decoder is added and groups the different subgroups corresponding to same word entities together.

2.3.3 Speech components and feature extraction

Speech Features

For years, physiology has inspired scientists to improve technology. Two techniques have been widely used for feature extraction in ASR since the 1980s: MFCCs (Davis & Mermelstein, 1980) and Perceptual Linear Prediction (PLP) analysis (Hermansky, 1990a).

MFCCs are a type of cepstral coefficient equally distributed on a mel-scale. A cepstrum is a non-linear transform with decorrelating properties first introduced by Bogert (1963). The mathematical equation of a cepstrum is described by following equation:

$$C_p = \left|\mathscr{F}^{-1}\left\{\log\left(|\mathscr{F}\{f(t)\}|^2\right)\right\}\right|^2 \tag{2.3}$$

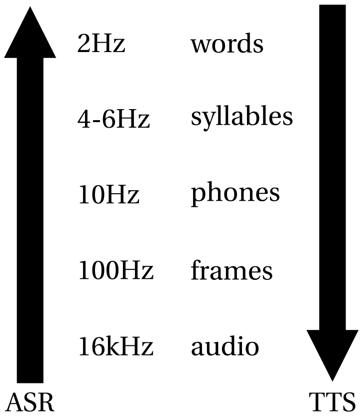


Figure 2.9: Frequency rates of different speech components.

Where $f(t)$ represents the signal and $\mathscr{F}$ represents the Fourier transform. The Mel scale is a subjective scale for the measurement of the pitch (Stevens et al., 1937).

The **PLP** analysis of speech estimates the auditory spectrum using the critical-band spectral resolution (by using the Bark scale transformation (J. O. Smith & Abel, 1999)), the equal-loudness curve (determining the sensitivity of hearing at different frequencies) (Robinson & Dadson, 1956) and the intensity-loudness power law (the non-linear relation between intensity and sound that can be estimated with a cubic root) (Stevens & Galanter, 1957). The acoustic model is approximated by an auto-regressive all-pole model (Hermansky, 1990a).

Speech components

Speech can be divided into different speech components at different frequency rates, as illustrated in Figure 2.9. The input to an ASR system can either be raw speech or preprocessed speech features. The output of the ASR system can be one of several options, including phonemes, syllabels or words. For ASR applications, speech is typically sampled at 16 kHz. Features are computed at a specific rate; standard MFCCs are computed out of a 25 ms window every 10 ms, which corresponds to the frame rate of 100Hz. Phonemes are typically sampled at a rate of 10 Hz, syllables around 4-6 Hz and words around 2 Hz.

The ASR system needs to be designed with these different sampling rates in mind.

2.3.4 Databases

Several databases are used in this thesis, adapted to the different experiments. The main datasets used are CHiME4, TIMIT, Librispeech and TIDIGITS, which are broadly used speech datasets for ASR. Besides for tasks that imply noise addition, we use the TIMIT dataset combined with the QUT-noise dataset with different SNR levels.

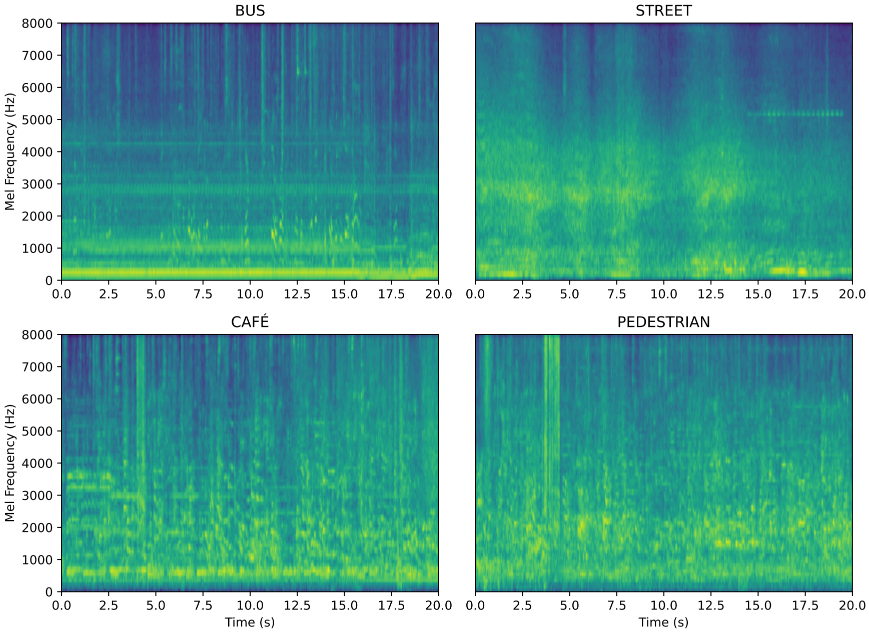


Figure 2.10: Summary of the different noises of the CHiME4 dataset.

CHiME4

The CHiME4 (Vincent et al., 2016) dataset is a noisy speech dataset based on the Wall Street Journal (WSJ) dataset (Paul & Baker, 1992). The clean part of the data consists of a combination of WSJ0 and WSJ1, which are well-established benchmarks for ASR. The noisy part of the dataset are divided in two parts: a simulated portion and a real portion. The simulated portion is created by artificially mixing clean utterances with noisy background noise, while the real portion consists of speech recordings made in different noisy environments. The noises used in this dataset for training, validation and testing are drawn from daily life environments such as buses, cafés, streets and pedestrian areas. The same utterances are recorded in multiple conditions to create a diverse dataset. A 20-second excerpt of each noise type used for simulation is shown in Figure 2.10. The bus noise is characterized by a persistent low-frequency component below 500 Hz, corresponding to the engine background noise. Similarly, the street noise contains low-frequency energy in the same range, primarily due to passing vehicles. In both environments, additional transient events, babble noise, and various everyday acoustic interferences are present. In contrast, the café and pedestrian area environments exhibit less low-frequency energy and are predominantly characterized by babble noise produced by surrounding individuals in close proximity to the speaker.

The data distribution is summarized in Table 2.1. In this thesis, we use the CHiME dataset to evaluate the modular capacity of a conformer-based network on different types of noisy speech in chapter 3.

Table 2.1: CHiME4 dataset summary with the number of utterances per subset. The noisy datasets contain the four types of noisy environments: bus, café, street and pedestrian area.

	clean	noisy simu	noisy real	total
train	37,416	42,828	9,600	89,844
dev	-	1,640	1,640	3,280
eval	1,206	1,320	1,320	3,846

TIMIT

The TIMIT dataset (Garofolo, 1993) is a well-established, manually annotated and relatively small dataset widely used in the ASR field used for phone recognition research (Lopes & Perdigao, 2011). The dataset contains 6300 utterances, each consisting of 10 sentences spoken by each of the 630 speakers recorded at a sampling rate 16kHz. The speakers are from eight major dialect regions of the US. About 33% of the speakers are female and 67% are males. Despite its relatively small size, TIMIT has been instrumental in supporting research into ASR systems for over three decades. This dataset is suitable for smaller-scale experiments aimed at developing or validating models. This dataset is also more suited for supervised learning. In this thesis, we use the TIMIT dataset on small ASR experiment notably in chapters 5 and 7.

Librispeech

The Librispeech dataset (Panayotov et al., 2015) is a widely used English dataset consisting of 1000 hours of audio recorded at a sampling rate of 16kHz. The data is derived from read LibriVox's audiobooks, with transcripts provided by the corresponding book texts. The dataset is divided into several subsets. The 'clean' subsets are particularly notable for their low Word Error Rates (WER) and closer to American English. The WER is the difference between the predicted text and transcripted text. In contrast, the 'others' subsets exhibit higher WER in preliminary ASR tests conducted by the authors. Librispeech is commonly used in the speech community for large-scale experiments and self-supervised learning tasks that require a substantial amount of data. The dataset's diverse range of speakers, accents, and speaking styles makes it an interesting choice for researchers seeking to develop or evaluate ASR systems. In this thesis, we use Librispeech for experiments that use transformer-based pretrained models in chapter 6.

Table 2.2: Summary of LibriSpeech dataset

Subset	**h**	**min./spk**	**f**	**m**	**tot.**	**exp.**
dev-clean	5.4	8	20	20	40	Evaluation
dev-other	5.3	10	16	17	33	Validation
train-100	100.1	25	125	126	251	SS and FT
train-360	263.6	25	439	482	921	SS
train-500	496.7	30	564	602	1166	SS

TIDIGITS

The TIDIGITS dataset (Leonard & Doddington, 1993) is a small dataset containing english spoken digit sequences. In this thesis, we use some utterances of TIDIGITS to illustrate the output of cochlear models on small spoken sequences.

QUT-NOISE

The QUT-NOISE corpus (Dean et al., 2010) is a background-noise dataset consisting of 20 recordings of 30 minutes of noise, recorded in diverse daily living environments: home, café, street, car, and reverberation areas. In this thesis, we use this corpus mixed with the TIMIT dataset to assess the noise robustness of our models trained on the TIMIT dataset.

2.3.5 Metric

In ASR, the primary metric used to evaluate the performance of a model is the Word Error Rate (WER) or Phone Error Rate (PER). WER measures the percentage of words in an utterance that are correctly predicted by an ASR system. It quantifies the difference between the transcriptions and the ground truth by dividing the sum of insertions (inserted words), substitutions (incorrectly replaced words) and deletions (missing words) by the total amount of words. A lower WER indicates a better performance of the model to transcribe spoken language. PER is a similar metric, but instead of the word-level, the percentage of error is based on phones.

WER is a standard benchmark used to objectively compare ASR systems, assess improvements due to updates and evaluate the performance on different datasets or different acoustic conditions.

3 Modularity

The concept of modularity in neural networks refers to a design principle that divides a model into several self-contained modules, each specialised for a specific action or with a specific structure. It is a concept that came up 30 years ago in neural networks, but recently found an interesting application area with the advent of large self-supervised pretrained models. The computation intensity of those models make it quite demanding resource-wise to train everything from scratch. By using modularity, several structures can be combined to solve complex problems more efficiently than big monolithic blocs.

Modularity also reflects the way biological systems such as the human neural network are built. Different regions of the auditory system specialize in distinct tasks: the cochlea extracts features, the auditory pathway refines these features and downsamples the signal, and the auditory cortex further processes the signal into interpretable neural signals be they words, syllables or phonemes. The main advantages of modularity are efficiency by decoupling a task in smaller modules, interpretability through specialized modules, transfer learning by importing one or more pretrained modules and scalability.

In the context of deep neural networks, modularity can be applied in three different ways:

- Architectural Modularity: Neural networks can be divided into subnetworks, focussing on specific functions. The inner architecture of different modules (CNN, recurrent neural network (RNN), transformers etc.) are chosen in function of the type of task the subnetwork has to perform. (LeCun et al., 2015)
- Functional Modularity: Inside a neural subnetwork, several modules can be used to accomplish different tasks. A typical example is multi-task learning. Each module has its own specialization (for example in ASR, there could be several languages), with the rest of the network shared. (Pfeiffer et al., 2023)
- Hierarchical modularity: Inside a network, different subnetworks specialize into recognizing specific attributes of the input. For example for speech a first module detects the activities at different frequencies, a second module combines those information

at different frequencies to detect phonemes and a last layer combines those phoneme information to understand words.

In the context of this thesis, modularity is applied to the ASR context introduced in the background chapter (Chapter 2) as illustrated in Figure 3.1. Notably, hierarchical modularity is applied with the idea of combining plausible cochlear models with existing deep learning structures and models. Hierarchical and architectural modularity will be further used in the chapters 5, 6 and 7.

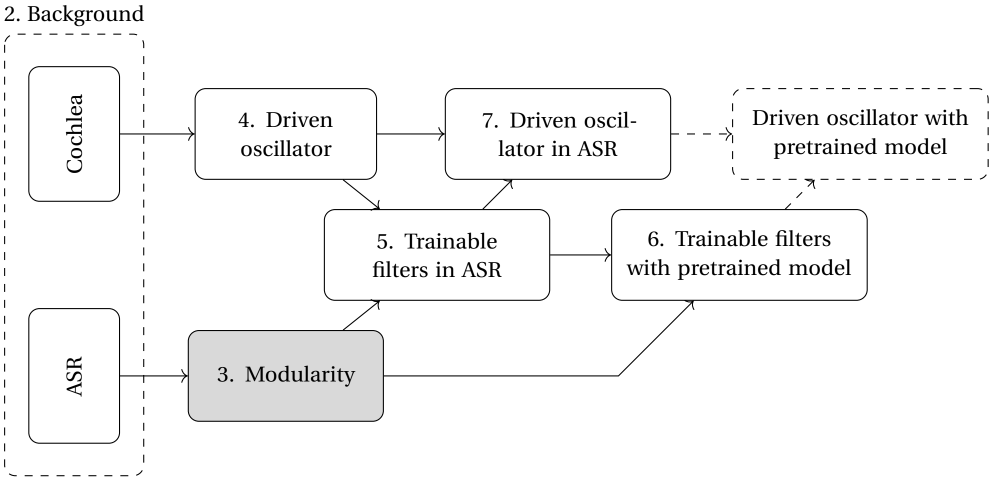

2. Background
Cochlea
ASR
4. Driven oscillator
7. Driven oscillator in ASR
Driven oscillator with pretrained model
5. Trainable filters in ASR
6. Trainable filters with pretrained model
3. Modularity

Figure 3.1: General overview of the thesis.

This chapter proposes a study done on an industrial internship which investigates more deeply how the concept of modularity has been used in the literature. This study focusses on functional modularity. A novel approach of applying a routing mechanism on a conformer-based system for doing ASR on noisy speech signals is proposed.

The majority of the text in this chapter is under revision, but available on *arXiv* as:

> Coppieters de Gibson, L., Garner, P. N., & Honnet, Pierre-Edouard(2024).An investigation of modularity for noise robustness in conformer-based ASR. *arXiv e-prints*

3.1 Introduction

With the advent of ever larger transformer-based architectures, the necessary computing power to train and infer with state of the art ASR models keeps increasing. Databases also grow in size and cover more modalities due to an increasing interest for multimodal and multitask modeling. This leads to an exponential increase of the number of model parameters (Sevilla et al., 2022) as well as amount of data required to train these models (Villalobos & Ho, 2022).

These larger models not only require large computational resources to be trained, they can also suffer from negative interference and catastrophic forgetting (Ramasesh et al., 2021). Inspired by biological systems (Sporns & Betzel, 2016), modularity has been applied in machine learning (ML) models for decades (Jacobs et al., 1991; Jordan & Jacobs, 1994). Recently, the concept of modularity has become popular again, especially for large models. Modularity can be introduced at different levels of a model and with different approaches, depending on the task and model structure (Pfeiffer et al., 2023). A modular structure allows to train only task related experts of a model instead of the whole model. Irrelevant experts for a given task are thus not trained, which saves computational power and avoids catastrophic forgetting. In the ASR field, mixture of experts (MoEs) have been shown to improve accuracy for different tasks by increasing the model size, while keeping the same computation power (K. Hu et al., 2023; You et al., 2021, 2022).

This work focuses on introducing modularity in conformer-based ASR models to handle speech in different types of noise environment. By adding modularity at the conformer block level, we allow the model to learn different conditions and exploit this information to improve its performance on both noisy and clean speech. We hypothesize that using different experts for different noisy types of speech will enhance the ASR performance of each type of noisy speech. Introducing modularity in the beginning of the model would make the model more robust to different types of noisy speech with fixed routing. We then explore learned routing, where we observe similar performance but with a model better suited to handle real-life situations. We demonstrate the effectiveness of the approach on the task of ASR in noisy environments. Using different experts for clean and noisy speech outperforms the standard conformer, however adding more granularity inside the noisy data class by separating the different types of noise does not improve the performance further. Another finding is that our modular models tend to be trained faster than the baseline conformer model without experts.

This study first reminds the theory behind modularity and conformers in the Section 3.2. Then Section 3.3 introduces our proposed method to tackle challenging noise environments for ASR. In Section 3.4, we report our experiments and findings. We conclude in Section 3.5.

3.2 Background

3.2.1 Modular networks

A modular neural network has three specific components: functional blocks or *experts*, a routing mechanism to select the right experts, and an aggregator that combines the outputs of those experts.

Functional blocks can be implemented in different ways:

- It can be a composition of parameters such as sparse subnetworks, where a small number of parameters are pruned to be trained for each specific task (Ansell et al., 2021) or low-rank modules such as LoRA (E. J. Hu et al., 2021).

- It can be obtained through an input composition where the input is concatenated with specific parameters (X. L. Li & Liang, 2021).
- It can be a function composition where a whole block is duplicated to act as different experts. (C. G. Rosenbaum, 2020)

A routing mechanism is needed to select an expert. This routing mechanism can either be fixed if the expert selection is known from the data (for example in multitask learning (Ruder, 2017)), or it can be learned when the routing information is not available. In case of learned routing, several challenges arise such as module collapse or training stability (C. Rosenbaum et al., 2019).

3.2.2 Conformer

The conformer architecture, introduced by Gulati et al. (Gulati et al., 2020), is a stack of conformer blocks. One such block is composed of two feedforward layers (one at the front and one at the end), one transformer layer and one convolutional layer. By design, it combines the advantages of both CNNs (T. N. Sainath et al., 2013) and transformers (Vaswani et al., 2017): CNNs primarily capture local contextual information and dependencies, while self-attention captures more global context.

3.3 Method

In this section we describe our two main contributions, namely the introduction of fixed routing and learned routing in the conformer architecture.

3.3.1 Fixed routing

Using a fixed routing mechanism implies knowing the condition in advance. When this is possible, a simple routing mechanism dictated by an input parameter can be set in place. This experimental setup gives us two keys: first it shows if using modularity to distinguish noisy and clean speech enhances the global performance. Second it can be used as pretrained model for a learned routing mechanism.

We propose to introduce modularity through experts at the conformer block level as illustrated in Figure 3.2: every expert in a modular layer is a full conformer block. To keep the same amount of computations between the baseline and the modular approaches, the router chooses exactly one expert for each utterance. The aggregator then composes the batch in the right order after the modular layer.

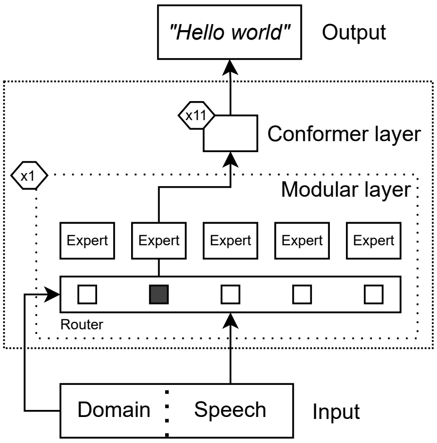


Figure 3.2: Fixed routing architecture

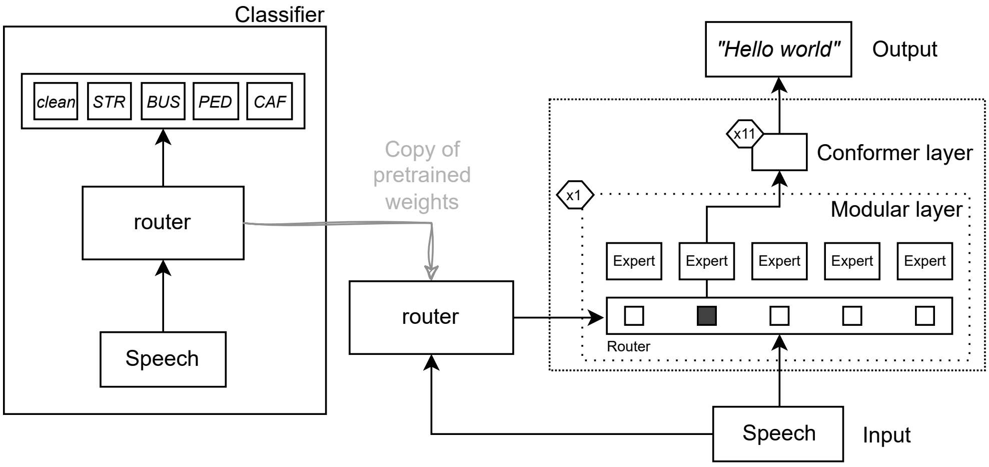


Figure 3.3: Learned routing architecture

3.3.2 Learned routing

The information about noise is not always available with the input. In daily situations, one can be exposed to outside noise or be in a noise free environment, but the model does not have access to the information. When this is the case, fixed routing cannot be used and the routing has to be inferred from the input signal.

Expert modules can only start to differentiate when the router has learned a consistent pattern. This gives two main paths to train a learned router with an ASR model: train everything together from scratch, or first pretrain the router before integrating it with the ASR.

Training everything together from scratch implies setting up a constraint to diversify the choice of the router output in the beginning of the training. This is needed to avoid a routing system that always picks the same expert, which forces the ASR experts to first learn a general solution, before starting to have a consistent routing.

The use of a pretrained router offers the advantage to be less computationally intensive, but it requires to create a parallel classification pipeline to pretrain the router. Pretraining the router avoids this forced diversification when training the ASR. Moreover, one can decide to freeze the router for some time while training the ASR and to unfreeze and train it jointly with the ASR in the next phase. In this paper, we used the second approach, illustrated in Figure 3.3.

3.4 Experiments

3.4.1 Dataset

Our experiments are carried out on the CHiME4 dataset presented in section 2.3.4.

3.4.2 Baseline and framework

Our implementation is based on the WeNet framework, an open-source toolkit used for streaming and non-streaming end-to-end ASR (Yao et al., 2021; B. Zhang et al., 2022). The baseline model is a 12-layer conformer encoder with a 6-layer transformer decoder. At every layer of the encoder, the attention module has 4 attention heads. The WER results of the baseline experiment are summarised in Table 3.1[I]. Two different decoders are used in the baseline experiment: a CTC beam search decoder and an attention rescoring decoder. A CTC Beam Search Decoder is a decoding algorithm used in end-to-end ASR. This method outputs a probability distribution out of a set of labels (i.e. characters) plus a special blank symbol (_) at every time step. The Beam search algorithm keeps track of the top N most likely word or utterance hypothesis at each time steps to find the most likely transcription (Graves et al., 2006). The attention rescoring algorithm uses the CTC beam seach decoding

---

[I] Note that the results reported by the authors on github differ from what we were able to reproduce, especially for the SE condition.

algorithm to generate a list of candidate transcriptions. Each transcription is then rescored using an attention-based decoder, taking into account the context of the sequence and the language model and picks the candidate transcription with the highes combined score (CTC and attention) (Chan et al., 2016; S. Kim et al., 2017). For the baseline we report both results, for the further experiments, we only report the attention rescoring decoder results, due to the better performance capacity.

Table 3.1: Baseline results: the results are computed for two different decoding methods: 'ctc' for ctc beam search and 'att' for attention rescoring. Five subsets are chosen: clean, real dev (RD), simu dev (SD), real eval (RE) and simu eval (SE) according to table 2.1.

	clean	RD	SD	RE	SE
ctc	17.73	20.91	22.48	30.85	53.66
att.	16.44	19.76	21.63	29.69	52.98

3.4.3 Fixed Routing

In the fixed routing experiment we explored the impact of modularity when using different numbers of experts and expert layers: The expert choices are:

For two experts, we also vary the number of layers which become modular: we experimented with 1, 2 and 3 modular layers. In addition, we also test the network behaviour when introducing the modularity only on the second or third layer rather than on the first layer of the network. The routing path is appended to the beginning of the input waveform to provide the router with the domain information.

Results are reported in Table 3.2. There exists an interesting trade-off between the number of experts that can be trained on specific data and the amount of data that each expert sees during training. If the data is diversified over the different experts, the more experts, the better the model will be adapted to that specific type of data. On the other hand if different types of data are too similar to be differentiated, the more experts, the less data each expert will receive

Table 3.2: Results of fixed routing with attention rescoring decoding method.

experts	mod. layer	clean	RD	SD	RE	SE
Baseline		16.44	19.76	21.63	29.69	52.98
2	1	10.02	**16.77**	19.93	26.28	27.33
2	1 - 2	10.43	16.98	19.74	26.22	27.25
2	1 - 2 - 3	10.16	17.99	20.30	26.69	28.03
3	1	10.47	18.12	19.63	27.21	27.76
5	1	10.30	17.46	19.89	26.82	28.25
2	2	10.18	16.99	**19.59**	**26.05**	**26.85**
2	3	**9.81**	16.99	19.63	26.27	26.88

Table 3.3: Number of utterances in each category for 2 experts

2 experts	clean	noise
train	37416	52428
dev	0	3280
test	1206	2640

Table 3.4: Number of utterances in each category for 3 experts

3 experts	clean	simulated noise	real noise
train	37416	42828	9600
dev	0	1640	1640
test	1206	1320	1320

to adapt its weights.

The results show that using modularity at the conformer block level outperforms the baseline both on clean and noisy speech. For clean speech, we achieve between 36.3% and 40.4% relative WER reduction, while for noisy speech the improvements lie between 6.1% and 15.1% for all types of noise except the simulated evaluation test set, where the baseline results differ from the rest of the results.

This means that specialising one layer to differentiate noise from clean environment enables the model to handle the two data types within different experts, which improves the general ASR performance. Going further into the implementation details, the results show that using 2 experts outperforms other settings most of the time on the CHiME4 dataset, which is probably linked to the trade-off discussed earlier.

3.4.4 Learned routing

The learned routing mechanism is divided into two parts: the router classifier and the ASR model (see Figure 3.3).

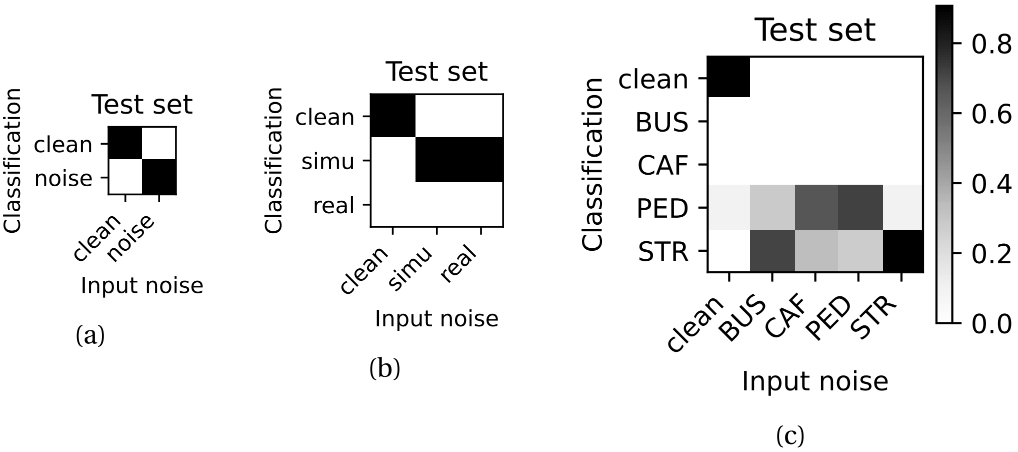
Router classifier

We first train a classifier to predict the target classes, which will become our pretrained router in the next stage. We opt for a simple architecture which consists of 3 CNN blocks. The goal of the router classifier is to predict the noise type of an input waveform. Since in the ASR model the conformer receives the waveform after the feature extraction, the router input can be the same. The different classes are the same as for our fixed routing experiments. An example for 5 experts is represented on the left side of Figure 3.3.

The confusion matrices obtained for the different classifiers are shown in Figure 3.4. For each figure, the true classes are given on the x-axis and the prediction is given on the y-axis

Table 3.5: Number of utterances in each category for 5 experts

5 experts	clean	pedestrian area	bus	street	café
train	37416	12768	13164	12990	13506
dev	0	820	820	820	820
test	1206	660	660	660	660



(a)

(b)

(c)

Figure 3.4: Confusion matrices for the different number of experts: (a) for 2 experts, (b) 3 experts and (c) 5 experts. The x-axis represent the type of domain we have at the input and the y-axis the output of the network.

of the matrix. For two experts (Figure 3.4 (a)), the two classes are clearly distinct. For three experts (Figure 3.4 (c)) the classifier is not able to make the distinction between real and simulated speech, but clean speech is clearly distinguished from noisy speech. Finally for five classes (Figure 3.4 (b)), the classifier is able to distinguish some noises, but there is still some confusion between the different types of noise. This means that the classifier is not able to separate the 4 different classes based on speech features. Interestingly, it groups some noises together: on the one hand 'human activities' (café and pedestrian area) and on the other hand 'car noises' (bus or street area) adds up as noise to the speech signal.

ASR

For the full ASR model with learned routing, the weights of the router classifier are loaded into the router part of the model (see Figure 3.3). We then did two different experiments: in the first one, we kept the weights of the router fixed during the whole ASR training, in the second one we kept the weights of the router fixed for the first five epochs and let the router then free to train.

The results, reported in Table 3.6, are similar to the ones obtained in the fixed routing experiments. This is due to the use of number of experts corresponding to what this dataset is able

Table 3.6: Results of learned routing mechanism

experts	number of fixed epochs	clean	RD	SD	RE	SE
Baseline		16.44	19.76	21.63	29.69	52.98
2	80	9.97	17.47	19.93	26.89	28.14
3	80	9.90	16.96	19.46	26.12	26.78
5	80	10.25	17.08	19.63	26.48	27.38
2	5	10.56	17.02	19.34	25.97	26.14
3	5	10.22	17.70	19.34	26.60	27.85
5	5	9.91	17.17	19.66	26.48	26.96

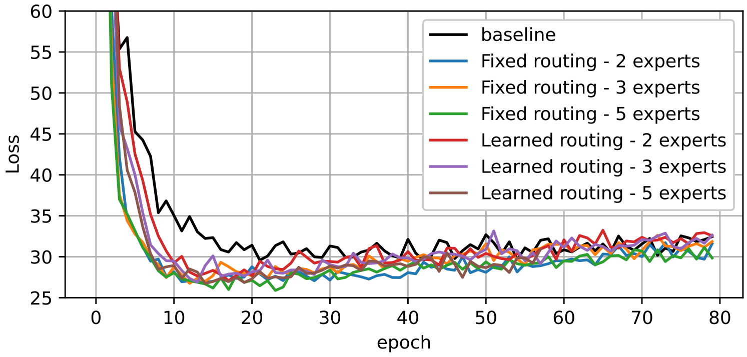


Figure 3.5: Loss function of different experiments: baseline and fixed and learned routing.

to differentiate amongst the different experts after feature extraction.

We then analysed what the router tended to learn when it is free to train. For 2 and 3 experts, after we unfreeze it, the router tends to transfer all the incoming data to the noise adapted expert, while for 5 experts the model keeps the different experts separated. The choice of the 'noise-robust' expert is probably due to the fact that this expert is trained to handle noisy speech and easily adapts to less noisy environments, while the other one only adapts to clean speech.

Further analysing the loss function (see Figure 3.5) shows that using routing helps the model to converge faster: after approximately 20 epochs, while the baseline experiment takes more time (about 25-35 epochs). This is reflected in the final model as we take the average of weights from the best 10 models, based on the validation loss. The final baseline model uses model checkpoints from epochs between 21 and 54, while for all the models where we introduce modularity, the final model is the average of checkpoints between the epochs 10 and 35[II]. The

---

[II] One exception was observed for the learned routing with 5 experts, with one outlier checkpoint being epoch 47.

implication is that we are able to reach better performance in a reduced training time and therefore less computing power. The loss curve however tends to increase after reaching a minimum. This can be due to overfitting to the training set or possibly to the dataset that does only have noisy data to validate the training after each epoch while the half of the training set consist of clean audio data.

3.5 Conclusion

In this study we examine the effectiveness of modularity on the CHiME4 dataset. We introduce modularity at the conformer block level and two routing options are explored: fixed and learned routing.

The fixed routing approach demonstrates that using modularity consistently outperforms the baseline across all conditions and configurations. However, dividing the data between two experts yields better results than using three or five. This suggests a trade-off: when input signals are distinctly different at the point where modularity takes place, using separate experts for each type improves overall performance. However, if the signals are similar, multiple experts may end up learning the same task, effectively reducing the data each expert processes.

We explored learned routing via a classifier-based router, which is pretrained before integration into the ASR system. This classifier shows that noisy speech is more challenging to differentiate after feature extraction, leading to a final division into two or three experts that distinguish between clean and noisy speech. This also points that noise distinction better works on signal-to-noise ratio (SNR) level than on the type of noise.

We also showed that the introduction of modularity allows for faster training, meaning reduced computational resources.

Future work may explore techniques to use fully learned routing without target classes. This approach can bring up other distinguishable elements helpful to ASR.

4 Hopf oscillator

In this chapter, we further elaborate the more accurate and complex model of the cochlea as introduced in the background chapter (Chapter 2) as illustrated in figure 4.1. The main goal of this chapter is to present a broad literature review of oscillator models and to present the prior work done in order to construct a Hopf module that can be integrated into an ASR structure in Chapter 7.

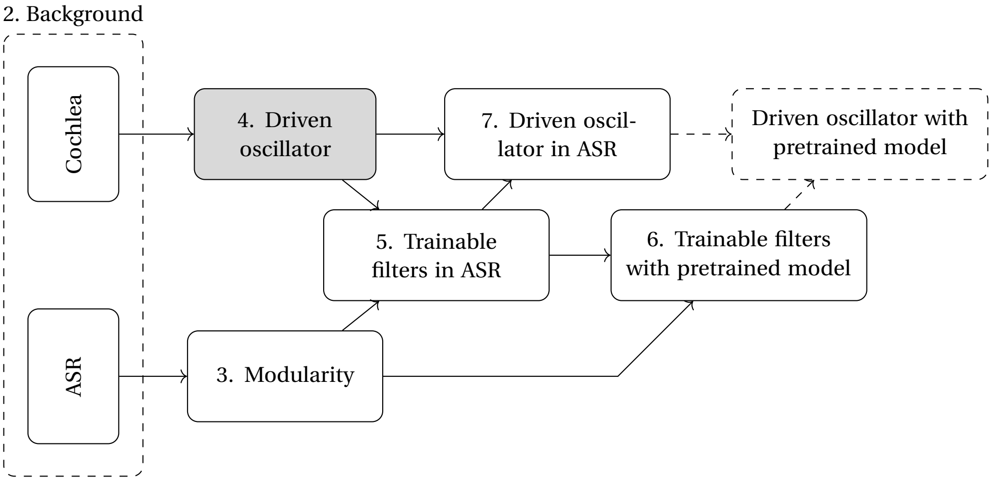

2. Background
Cochlea
ASR
4. Driven oscillator
7. Driven oscillator in ASR
Driven oscillator with pretrained model
5. Trainable filters in ASR
6. Trainable filters with pretrained model
3. Modularity

Figure 4.1: General overview of the thesis.

To a first approximation, the cochlea can be modelled as a filterbank. However, a better understanding of the underlying biological mechanism rather presents the cochlea as an array of active amplification oscillators poised at the Hopf bifurcation.

This active amplification mechanism is due to a neural feedback mechanism that actively amplifies the haircell movements when the acoustic signal amplitude is lower than a given threshold and damps it down when the amplitude is higher.

This mechanism implies two specific elements:

- There is a cube root compression on the amplitude level between the acoustic amplitude level and the signal transmitted in the inner ear.

- An active adaptation takes place when switching between different input signal amplitude levels.

This mechanism can be modelled by a set of differential equations which can then be integrated into a recurrent module.

This chapter starts with some background information about oscillators and the Hopf mechanism. Firstly, we give a brief overview of the mathematical expression of harmonic oscillator and the Hopf oscillator is presented with an explanation of the intricacies of the different parameters. Then, a broad review of the cochlear models based on oscillators in the literature is presented, highlighting the diversity of formulation that have emerged through different papers. A 'take-away' section summarizes the main key points that we want to integrate into our own model. The bifurcation being an important notion in the Hopf model, a broad review of the different types of bifurcations is proposed and the understanding of the Hopf oscillator is detailed. Further, we present a simulation, demonstrating the adaptation capabilities of an array of oscillators to external signal. A last section details the results of ASR experiments we did using classical MFCCs and CARFAC features. The main goal of this section is to compare physiologically plausible features incorporating active gain control with classical MFCC features in terms of performance.

4.1 Background

This section goes into mathematical details of oscillator-based cochlear models in the literature, skipping this section would not compromise the understanding of the rest of this chapter. To understand the main elements that we retain from the literature however, subsection 4.1.1 introduces the Hopf oscillator and the implications of a bifurcation in the context of this oscillator and subsection 4.1.4 provides a good summary of the takeaways of this section.

4.1.1 Types of oscillators

There exist two main expressions of the oscillators in the context of the cochlea : one based on the general driven harmonic oscillator form and another based on the normal form of the Hopf bifurcation equation implying a complex form and a cube root non-linearity.

The harmonic oscillator

The harmonic oscillator is the common second order oscillator. A simple harmonic oscillator (neither damped or driven) results from a combination of the 2nd Newton Law ($F = ma$) and

Hooke's law for a mass on a spring ($F = kx$).

$$m\ddot{x} + kx = 0 \tag{4.1}$$

In this equation, m corresponds to the mass of an oscillating object, x is its position and k is the stiffness, a constant related to the spring. In real-world applications, oscillators are often damped or driven. Mathematically, the damping behaviour can be expressed by adding a friction component to equation 4.1. This component is proportional to the velocity $\dot{x}$ and multiplied by a viscous damping coefficient h. Coupled to an external force $F(t)$, oscillators can also be driven.

$$m\ddot{x} + h\dot{x} + kx = F(t) \tag{4.2}$$

Hopf bifurcation

The Hopf bifurcation arises from an oscillator function that naturally has two types of regime: a damping regime and an active amplification regime. The point where the oscillator switches from one regime to the other is called the bifurcation point. The normal form of the Hopf oscillator is given by the following differential equation:

$$\dot{z} = z(a + b|z|^2) \tag{4.3}$$

The variables z, a and b are complex numbers and can be rewritten as $z = re^{i\theta}$, $a = \mu + \omega_0 i$ and $b = \beta + i\gamma$ where β represents the first Lyapunov coefficient. This coefficient should be negative to have stable solutions for any real part of a. Equation 4.3 can be written as:

$$\dot{r}e^{i\theta} + i\dot{\theta}re^{i\theta} = re^{i\theta}(\mu + \omega_0 i + (\beta + i\gamma)r^2) \tag{4.4}$$

Separating the real and imaginary part, we get two differential equations, this system of equations is the normal form of the complex Hopf oscillator model used throughout this thesis.

$$\begin{cases} \dot{r} = r(\mu + \beta r^2) \\ \dot{\theta} = \omega_0 + \gamma r^2 \end{cases} \tag{4.5}$$

The stable oscillating solution is obtained by setting $\dot{r} = 0$. The oscillator stops any oscillation ($r = 0$) or converges to a stable limit cycle with a fixed amplitude ($r = \pm\sqrt{\frac{-\mu}{\beta}}$). For a negative value for β, the regimes will depend on the value of μ (see Figure 4.2).

- $\mu \leq 0$: the equation has one solution: a stable fixed point in 0, oscillations will tend towards 0

- $\mu > 0$: the equations has 3 solutions: one unstable fixed point in 0 and 2 stable solutions in $\pm\sqrt{\frac{-\mu}{\beta}}$. If we consider the radius as being strictly positive, the the oscillator will tend towards a stable limit cycle with a radius equal to $\sqrt{\frac{-\mu}{\beta}}$.

The point where the solution switches from a single stable point to a stable limit cycle is called the bifurcation point. This bifurcation point is the limit between an active amplification mode (stable limit cycle) and a damping mode. This bifurcation point is a key concept in understanding of the working of the cochlea.

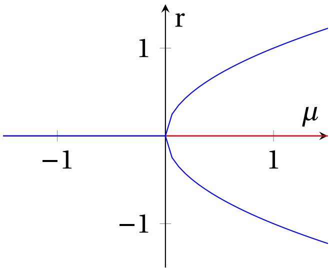


Figure 4.2: Schematic of the solutions of equation 4.5: the radius obtained in function of the bifurcation parameter (μ). The bifurcation occurs at $\mu = 0$.

In order to facilitate the understanding of the above literature, figures 4.3, 4.4 and 4.5 can help understand the purpose of active oscillation.

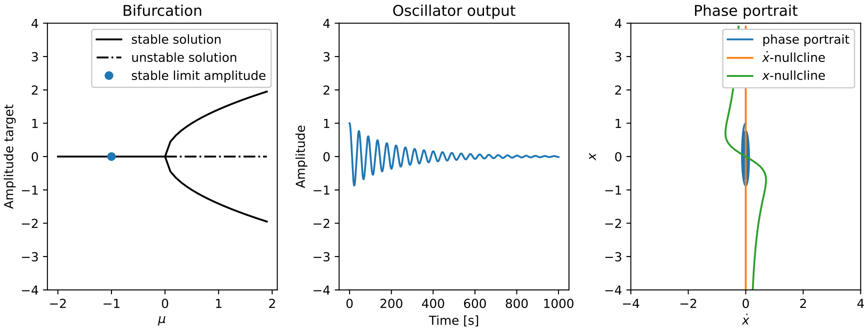


Figure 4.3: Damping regime

The figures are organized as follows: On the left, the bifurcation plot illustrates the target amplitude with respect to the bifurcation parameter (μ). In the middle, the real part of the oscillator ($x = r\cos\theta$) output linked to the bifurcation parameter. On the right the phase portrait of the response with the nullclines of x and $\dot{x}$. The phase portrait represents the orbits of the oscillator output in the phase space. It turns counterclockwise, and its dynamic is dictated by the nullclines. Nullclines correspond to the points where the associated derivative is equal to 0, it is vertically oriented when crossing the x-nullcline and horizontally oriented when crossing the $\dot{x}$-nullcline. The x-nullcline, dependent of the bifurcation parameter will dictate a damping or amplification regime. The oscillator output evolves towards the target solution of the bifurcation plot. If the bifurcation parameter is below zero, the amplitude is damped down. This means that the acoustic input is higher than the threshold. In figure 4.4

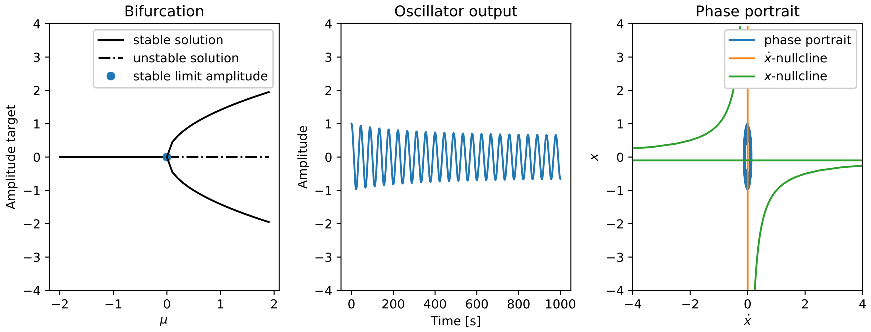


Figure 4.4: Hopf bifurcation regime

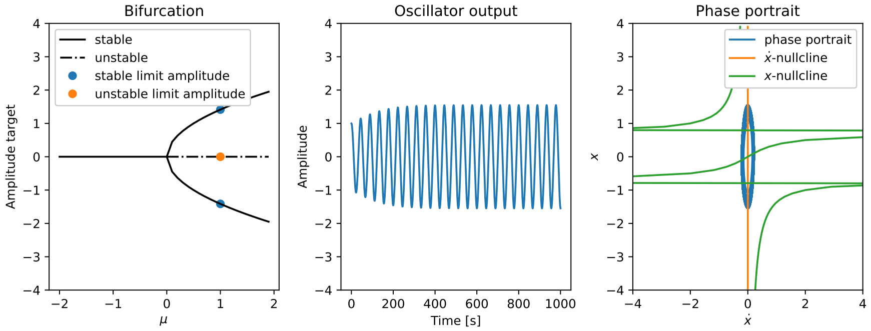


Figure 4.5: Active amplification regime

we are at the bifurcation point. The amplitude is still damped down, but to get to zero we would need an infinite amount of time. The acoustic input is at the threshold. Finally, in figure 4.5, μ is above zero, the amplitude is amplified. This means that the acoustic input is lower than the threshold, the Hopf mechanism actively tries to amplify the incoming signal.

4.1.2 Cochlear model equations in the literature

There exist several types of differential equations describing the working of the cochlea. However out of all the proposed mathematical models, two main categories are evident. The first category proposes to describe the oscillation of the cochlea by means of a harmonic oscillator. This describes the position of the haircell with respect to its acceleration, velocity and vibration force (Duke & Jülicher, 2008; Gianoli et al., 2017, 2022; Nobili et al., 1998). The second category places the bifurcation as central in the equation and proposes a complex-form differential equation. This approach describes the oscillation in function of the radius and the frequency of the haircell movement. Further some combinations of the two equations are proposed by some authors by introducing an active amplification force into the harmonic oscillator equation.

In order to make the reading easier, we changed some variable names of the equations taken from the literature in this section to make the notation correspond to the variables introduced in Section 4.1.1.

Haircell position equation

Fluid-coupled oscillator implementations based on the harmonic oscillator (equation 4.2) are used in cochlear models to describe oscillations in the cochlea.

Considering an array of oscillator, each oscillator can be described using the harmonic equation with an index i:

$$m_i \ddot{x}_i + h_i \dot{x}_i + k x_i = F_i(t) \tag{4.6}$$

Nobili et al. (1998) proposes to model the cochlea with two oscillator arrays. The first fluid-coupled harmonic oscillator models the basilar membrane oscillations. Taking the harmonic oscillator equation (equation 4.6) as baseline, they propose a more complete description in the physiological sense. The force driven at a specific frequency i, $F_i(t)$ is generated by the acceleration $a_s(t)$ of the ossicles in the middle ear when transmitting the sound vibration to the inner ear. It is transmitted by the cochlear fluid to oscillator i. Which can be written as:

$$F_i(t) = -G_i a_s(t) \tag{4.7}$$

Three terms can be added to the harmonic oscillator equation (4.6):

- The hydrodynamic term ($G_i^j \ddot{x}_i$) represents the force caused by oscillator j transmitted to oscillator i.
- The shear viscosity term $\big(s_i(2\dot{x}_i - \dot{x}_{i-1} - \dot{x}_{i+1})\big)$ represents the viscous forces acting on oscillator i which are influenced by the velocity of the neighbouring oscillators
- The force to oppose damping ($U_i(y_i)$) which are generated by the fast OHC contractions.

Combining the harmonic oscillator equation (4.6) with those different component leads to the following equation:

$$\sum_{j=1}^{N} (G_i^j + m_i \partial_j^i)\ddot{x}_i + h_i \dot{x}_i + s_i\big(2\dot{x}_i - \dot{x}_{i-1} - \dot{x}_{i+1}\big) + U_i(y_i) + k x_i = -G_i a_s(t) \tag{4.8}$$

The basilar mebrane movements induce a direct movement of the stereocilia. Through stereocilia displacements y_i, the tectorial membrane forms a second array of oscillators that is sparsely coupled to the basilar membrane x_i. The equations of motion of the second oscillator can also be written as a harmonic oscillator equation with the force being the coupling to the basilar membrane ($-C_i\ddot{x}_i$).

$$\bar{m}_i\ddot{y}_i + \bar{h}_i\dot{y}_i + \bar{k}y_i = -C_i\ddot{x}_i \tag{4.9}$$

Where $\bar{m}$, $\bar{h}_i$, $\bar{k}$ and $C_i$represent mass, damping, stiffness and coupling constant respectively. At resonance, the first and third terms at the left hand side of the above equations cancel. After time integration, the relation between the hair cell position and basilar membrane velocity becomes:

$$y_i = -\frac{C_i}{\bar{h}_i}\dot{x}_i \tag{4.10}$$

Therefore, at resonance, the OHC force term $U_i(y_i)$, in the linear approximation, behaves like a negative viscosity term and undamps cochlear motion.

Gianoli et al. (2017, 2022) propose a modelling of the hair bundle of the OHC by harmonic oscillator. The force balance on the hair bundle is described as:

$$m_{HB}\ddot{x} = -h_{HB}\dot{x} - k_{SP}(x - x_{SP}) - \sum_{j=1}^{N_t} F_t^j \tag{4.11}$$

With m_{HB} the hair bundle's apparent wet mass, h_{HB} the bundle's viscous drag coefficient, k_{SP} the combined stiffness of the stereociliary pivots, N_t the number of tip links, x represents the position of the hair bundle, $\dot{x}$ and $\ddot{x}$ are it's first and second order derivatives and x_{SP} stands for the resting position. For the implementation of the second order equation, (Gianoli et al., 2022) separates equation 4.11 in two first order equations by setting $y = \dot{x}$.

$$\begin{cases} \dot{y} = -h_{HB}y - k_{SP}(x - x_{SP}) - \sum_{j=1}^{N_t} F_t^j \\ \dot{x} = y \end{cases} \tag{4.12}$$

This way of describing the elongation of the harmonic oscillator is similar in both examples and this type of equation is mainly used to describe the position of the hair bundle. No bifurcation is part of those equations. In order to get a bifurcation point, a third-order non-linearity should be added.

Duke and Jülicher (2008) also propose a harmonic oscillator system to describe hair bundles. An active force term $F_a(x,\dot{x})$ is introduced, which gathers all forces linked to the inner structure of the hair cells.

$$m\ddot{x} + h\dot{x} = F_s + F_a(x,\dot{x}) \tag{4.13}$$

This active force makes the link with the amplification mechanism induced by the outer hair cells. In all the harmonic oscillator based implementations, an extra term is added to model this interaction with the OHC. Those terms are non-linear if they want to capture the cube root compression mechanism present in the cochlea.

Complex Hopf oscillator equation

The other classical implementation of cochlear models is based on Hopf bifurcation implementations. This implementation rather focuses on integrating the non-linear aspect of the cochlea: the active amplification by the OHCs resulting in OAEs when the input signal of the cochlea is below a given threshold. The literature relates to equations 4.3 and 4.5.

Camalet et al. (2000) introduced the concept of self-tuned Hopf bifurcation as model for the cochlea. He started of with the basic pitchfork bifurcation equation, writing the variables a and b as complex variables depending on a tuning parameter μ and the frequency ω.

$$f_1 = a(\omega,\mu)z_1 + b(\omega,\mu)|z_1|^2 z_1 + \dots \tag{4.14}$$

In this paper, a simple self-tuning equation was proposed to tune the μ parameter in function of the incomming signal:

$$\frac{1}{\mu}\dot{\mu} = \frac{1}{\tau}\Big(\frac{z^2}{\delta^2} - 1\Big) \tag{4.15}$$

This equation easily tunes the equation towards the bifurcation point and can also head away from it. This mechanism reflects in a mathematical way the more complex biochemical processes in the hair cells involving the dynamics of Ca^{2+}-channels and molecular interaction.

Hudspeth et al. (2010b) further develops the equation from Camalet et al. (2000) by explicitly using $a = \mu - \mu_C - i\omega$, where μ_C stands for the critical value of the Hopf bifurcation (when $\mu = \mu_C$) and ω_C is the frequency at which the oscillator would naturally oscillate when being in active amplification mode.

$$\dot{z} = -(\mu - \mu_C - i\omega)z - bz|z|^2 + F \tag{4.16}$$

J. C. Kim and Large (2015) considers a version of the Hopf oscillator equation proposed by Large et al. (2010) to describe the model of the cochlea. This equation takes higher order terms into account the interaction between oscillators through a connectivity parameter ϵ which is a small real number. In this equation β is notated as β_1 and corresponds to the first Lyapunov coefficient ($l_1 = \beta_1$). For the extra term β_2 is introduced which is directly related to the second Lyapunov coefficient ($l_2 = \epsilon\beta_2$). Those coefficients determine the autonomous behaviour of the canonical Hopf oscillator: the number of stable solutions and bifurcations.

$$\dot{z} = z\Big(\mu + i\omega + \beta_1|z|^2 + \frac{\epsilon\beta_2|z|^4}{1-\epsilon|z|^2}\Big) + F(t) \tag{4.17}$$

In order to adapt the frequency range of each oscillator to the non-linear range perception of frequencies in the cochlea, a frequency-scaling factor f is introduced. High frequency oscillators have a larger bandwidth than low frequency oscillators.

$$\frac{1}{f}\dot{z} = z\left(\mu + \mathrm{i}\frac{\omega}{f} + \beta_1|z|^2 + \frac{\epsilon\beta_2|z|^4}{1-\epsilon|z|^2}\right) + F(t)e^{\mathrm{i}\omega_0 t} \tag{4.18}$$

Considering the polar form this can be divided in the following set of equations disentangling the radius and frequency of the oscillators by setting $z = re^{i\theta}$. The derivation to obtain the following equation is similar to the development in equation 4.4.

$$\begin{cases} \frac{1}{f}\dot{r} = \mu r + \beta_1 r^3 + \frac{\epsilon\beta_2 r^5}{1-\epsilon r^2} + F\cos\theta \\ \frac{1}{f}\dot{\theta} = \frac{\omega}{f} - \frac{F}{r}\sin\theta \end{cases} \tag{4.19}$$

The frequency-scaling is also used by Stoop et al. (2016), who is mainly interested in understanding in combination-Tone Laws with the normal Hopf equation form.

Critics, combinations etc.

Some research studies have aimed to bridge the gap between the harmonic oscillator equations and Hopf bifurcation theory. By exploring intermediate equations, these studies aim to integrate insights from both paradigms.

Duifhuis (2011) conducted a mathematical analysis and compared simulations of van der Pol, Rayleigh, and Hopf oscillators. They concluded that while Hopf-bifurcation critical oscillators cannot model OAEs, van der Pol oscillators are capable of doing so. Furthermore, their study found that van der Pol oscillators can exhibit chaotic behavior, whereas critical oscillators produce a stable response. However, their analysis did consider all parameters were fixed. Considering the bifurcation parameter (μ) as trainable dependent on the external signal amplitude, could enable OAEs within a Hopf bifurcation model, as proposed by Camalet et al. (2000).

Hudspeth et al. (2010b) proposed a link between the Hopf equation 4.2 and the harmonic oscillator equation 4.3. Starting from the harmonic oscillator form, a non-linear undamping force $\bar{U}_n(\dot{x}_n)$ is added, which occurs at resonance.

$$m_n\ddot{x}_n + h_n\dot{x}_n + k_n x_n + \bar{U}_n(\dot{x}_n) = F_n \tag{4.20}$$

$$\bar{U}_n(\dot{x}_n) \cong -h'_n\dot{x}_n + \alpha_n\dot{x}_n^2 + \beta_n\dot{x}_n^3 \tag{4.21}$$

When considered at the Hopf bifurcation, the combination of those different equations can be brought to a general equation of type:

$$\bar{F} = ax + b|x|^2 x \tag{4.22}$$

Which corresponds to the general Hopf bifurcation equation. By setting:$x = \bar{x}e^{i\omega t}$ if we consider that $\bar{x}$ and ω are not dependent on time. Then a and b would be defined as :

$$\begin{aligned} &a = k_n - m_n\omega^2 + i\left(h_n - h'_n\right)\omega \text{ and} \\ &b = \left(4\alpha_n^2\omega^4\right) / \left[k_n - 4m_n\omega^2 + 2i\left(h_n - h'_n\right)\omega\right] + 3i\beta_n\omega^3 \end{aligned} \tag{4.23}$$

This expressions of a and b make the link between the two families of oscillators.

Duke and Jülicher (2008) also manage to draw a link between the harmonic oscillator implementation and the Hopf oscillator equation. They propose the implementation of the harmonic oscillator with an active force term f_a.

$$m\ddot{x} = -\lambda\dot{x} - kx + F_a + F \tag{4.24}$$

This active force corresponds to the force obtained by the active amplification mechanism of Hopf oscillators. Several mathematical expressions of this active force are proposed:

- The active force can diminish with amplitude:

  $$F_a = (C - Bx^2)\dot{x} \tag{4.25}$$

- The active force can diminish with velocity:

  $$F_a = (C - B\dot{x}^2)\dot{x} \tag{4.26}$$

- The active force can be proportional to the displacement with a given time delay

  $$F_a = Cx(t - \tau) \tag{4.27}$$

- If we consider the inertial effects as negligeable $(m\ddot{x} = 0)$, the stiffness as nonlinear $(k = k(x))$ then, if f_a evolves as a first-order differential equation, we end up with the following equation set (the mathematical details are given in appendix A.1:

  $$\begin{cases} \lambda\dot{x} = -(k - C + Bx^2)x + F_a + F \\ \tau\dot{F}_a = -F_a - \bar{k}x \end{cases} \tag{4.28}$$

4.1.3 Criticality concept

The Hopf bifurcation relates to the criticality concept, this concept refers to the property of some complex systems to operate near a critical point (CP) between order and chaos. Self-organized criticality was introduced by Bak et al. (1988) which describes how some systems inherently evolve near this CP.

In physics, critical points are related to phase transitions (Stoop & Gomez, 2022). Typical examples in nature are the transition phase between liquid-gas phases for liquids, an epidemic threshold or superconductors. The critical point is the point where a system will be in-between a diverging and converging state. In the context of the Hopf oscillator, the critical point occurs when the solution changes from a damping to an active amplification regime.

Systems working near a critical point are characterized by critical exponents, which characterize how a system or physical quantity diverges or vanishes near a critical point. In the brain this equilibrium is found between excitatory and inhibitory neurons and at the critical point you can define avalanches of excitatory behaviour, allowing an efficient transmission of information across different neuron layers.

According to Munoz (2018), life would have evolved around this criticality point. A lot of physiological elements such as sensory systems, neural connections in the brain, genes and stem-cells. The main advantages of a critical system are the optimal transmission capacity (without vanishing or exploding), the optimal information processing, the largest repertoire for memory storage and the optimal sensitivity and dynamic range to incoming stimuli (at the critical point, the dynamic range is the most diversified).

There exists a whole zoo of criticality equations, depending on the application (Bedi et al., 2015; Kinouchi & Copelli, 2006). In this thesis we concentrate on the Hopf bifurcation equation but understanding that this concept of being poised near the critical point is a common behaviour in physiological systems.

4.1.4 Take aways for building our model

The mathematical approach that effectively integrates the bifurcation mechanism is the Hopf equation, which has been shown in several papers to successfully mimic the functioning of hair cells within the cochlea. From them, two concepts from the literature are also considering in our model implementation:

1. The frequency scaling introduced in J. C. Kim and Large (2015)

   $$\left\{ \begin{array}{l} \frac{1}{f}\dot{r} = \mu r + \beta_1 r^3 + \frac{\epsilon\beta_2 r^5}{1-\epsilon r^2} + F\cos\theta \\ \frac{1}{f}\dot{\theta} = \frac{\omega}{f} - \frac{F}{r}\sin\theta \end{array} \right. \tag{4.29}$$

   This equation adapts properly to the logarithmic scale of a classic filterbank as shown in figure 4.6. The oscillator bandwidths are scaled around central frequencies which are evenly spaced on a logarithmic scale.

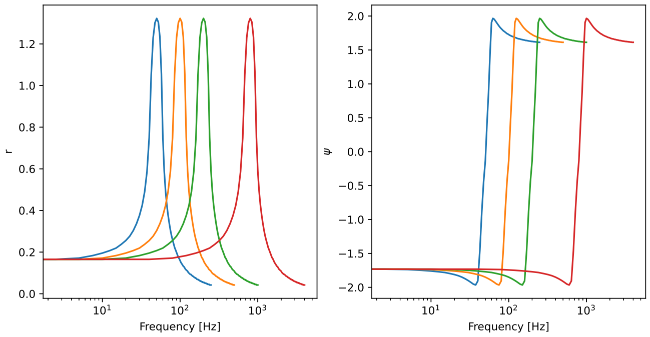


Figure 4.6: Oscillator filterbank after frequency adaptation.

2. The self-tuning parameter introduced by Camalet et al. (2000):

$$\dot{\mu} = \frac{\mu}{\tau}\left(\frac{x^2}{\delta^2} - 1\right) \tag{4.30}$$

   This equation is a simplified version of the real molecular physics that are going on in the hair cells. It adds a dependency of the bifurcation parameter (μ) with the amplitude of the incoming signal.

4.2 Bifurcations

The cochlea is tuned on the edge of a Hopf bifurcation (Hudspeth et al., 2010b). A bifurcation is a change in the number of solutions to a differential equation. This section further delves into different bifurcations in the literature. We show that the oscillators in the cochlea combine two types of bifurcations.

4.2.1 Fold bifurcation

The fold bifurcation, also called saddle-node bifurcation, is a bifurcation where the solution consists of two fixed points (a stable and an unstable fixed point) which collide and annihilate each other at the bifurcation. The bifurcation is induced by the variation of an external force or constant variable. The typical example of the fold bifurcation is given by equation:

$$\dot{r} = -r^2 + F \tag{4.31}$$

When varying F, the bifurcation point occurs when $F = 0$. Figure 4.7 shows how the derivative evolves in function of F and figure 4.8 illustrates how the stable and unstable solution collide

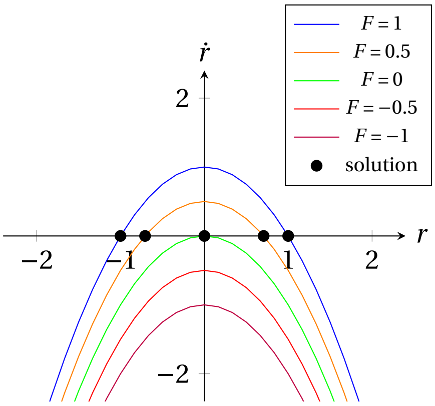


Figure 4.7: Evolution or $\dot{r}$ in function of r for different values of F. The solutions of the differential equation vary between 0,1 and 2 solutions.

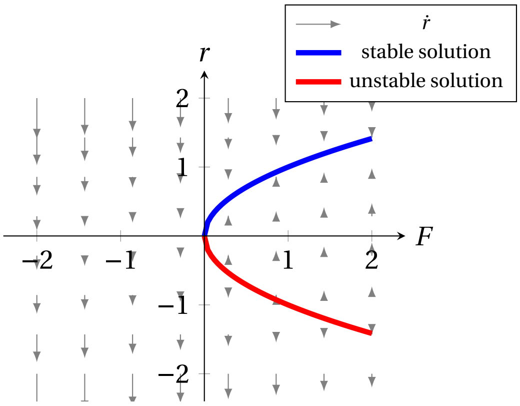


Figure 4.8: Schematic of the solutions on the r-F plane. The stable and unstable nullclines are indicated in blue and red respectively.

when $F = 0$. The solutions are given by:

$$F < 0 \quad : \quad \emptyset \tag{4.32}$$

$$F \geq 0 \quad : \quad r = \pm\sqrt{F} \tag{4.33}$$

4.2.2 Transcritical bifurcation

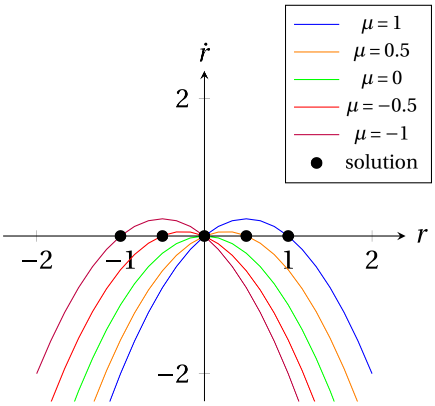


Figure 4.9: Evolution or $\dot{r}$ in function of r for different values of μ. The solutions of the differential equation vary between 1 and 2 solutions.

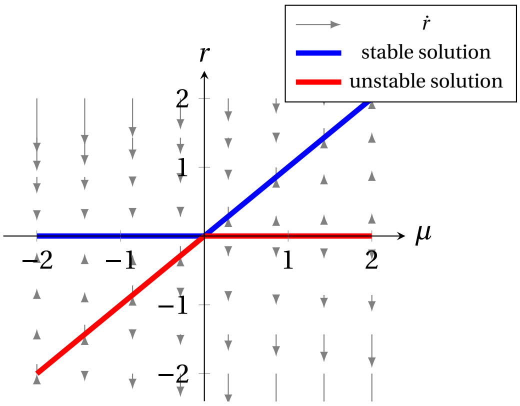


Figure 4.10: Schematic of the solutions on the r-μ plane. The stable and unstable nullclines are indicated in blue and red respectively.

The transcritical bifurcation is a bifurcation with two fixed points as solution where the stability between the two fixed points is exchanged at the bifurcation. This bifurcation is induced when varying the μ parameter. The typical example of the transcritical bifurcation is given by equation:

$$\dot{r} = \mu r - r^2 \tag{4.34}$$

When varying μ, the bifurcation point occurs when $\mu = 0$. Figure 4.9 shows how the derivative evolves in function of μ and figure 4.10 illustrates how the stable and unstable solution exchange their stability at $\mu = 0$. The solutions are:

$$\mu < 0 \quad : \quad r = \mu, r = 0 \tag{4.35}$$

$$\mu = 0 \quad : \quad r = 0 \tag{4.36}$$

$$\mu > 0 \quad : \quad r = 0, r = \mu \tag{4.37}$$

4.2.3 Pitchfork bifurcation

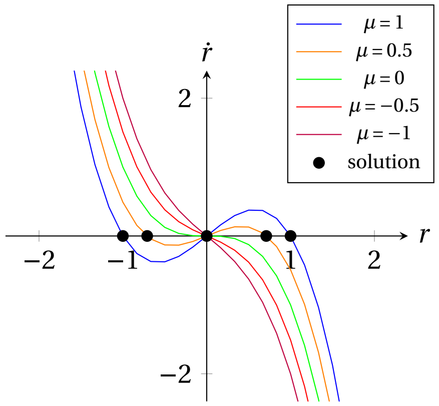


Figure 4.11: Evolution or $\dot{r}$ in function of r for different values of μ. The solutions of the differential equation vary between 1 and 3 solutions.

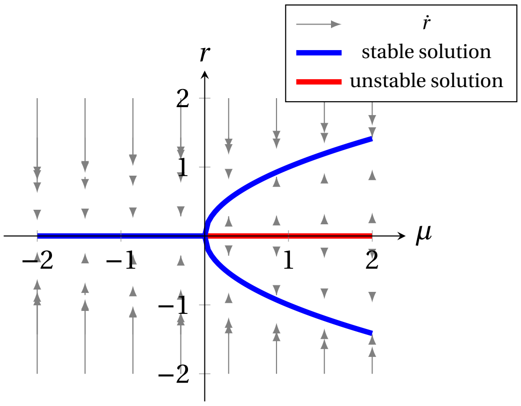


Figure 4.12: Schematic of the solutions on the r-μ plane. The stable and unstable nullclines are indicated in blue and red respectively.

The pitchfork bifurcation is a bifurcation with up to three fixed points as solution. This type of bifurcation combines the phenomena of the fold and transcritical bifurcations. At the bifurcation point, two solutions collide going from three to one solution, and the stable solution changes from stable to unstable. The typical example of the pitchfork bifurcation is given by the following equation:

$$\dot{r} = \mu r - r^3 \tag{4.38}$$

When varying μ, the bifurcation point occurs when $\mu = 0$. Figure 4.11 shows how the derivative evolves in function of μ and figure 4.12 illustrates how the stable and unstable solution collide at the bifurcation. The solutions are given by:

$$\mu < 0 \quad : \quad r = 0 \tag{4.39}$$

$$\mu = 0 \quad : \quad r = 0 \tag{4.40}$$

$$\mu > 0 \quad : \quad r = 0, r = \pm\sqrt{\mu} \tag{4.41}$$

4.2.4 Hysteresis bifuration

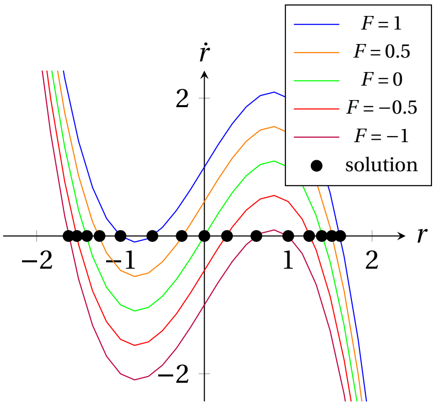


Figure 4.13: Evolution or $\dot{r}$ in function of r for different values of F. The solutions of the differential equation vary between 1,2 and 3 solutions.

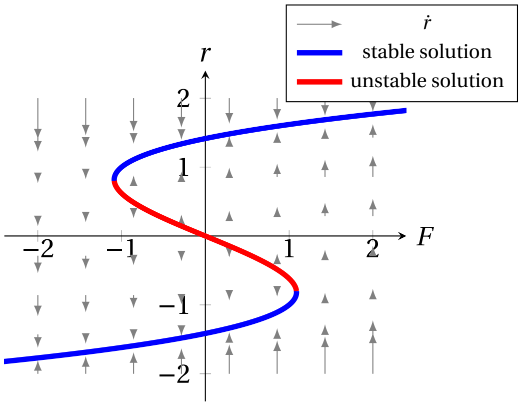


Figure 4.14: Schematic of the solutions on the r-F plane. The stable and unstable null-clines are indicated in blue and red respectively.

The hysteresis bifurcation is a bifurcation with up to three solutions. It mathematically combines the fold and transcritical bifurcations by adding the μ and F parameters. However, only the F parameter is considered as a varying parameter. The hysteresis bifurcation has two bifurcation points: at each of these points, a stable and unstable solution collide or appear, while the third solution continues. The typical example of the hysteresis bifurcation is given by equation:

$$\dot{r} = F + \mu r - r^3 \tag{4.42}$$

The solutions are defined by an inverted third-order equation, which is mathematically difficult to invert. Figure 4.13 shows how the derivative evolves in function of F and figure 4.14 illustrates how the stable and unstable solutions collide at the two bifurcation points. A bifurcation occurs only when $\mu > 0$, switching from one solution to three, so the following categories can be drawn:

- If $\mu < 0$: one fixed point, for the whole range of r (there is no bifurcation)
- If $\mu > 0$:

- $r < -\sqrt{\mu}$: one stable fixed point
- $r > \sqrt{\mu}$: one stable fixed point
- $-\sqrt{\mu} < r < \sqrt{\mu}$: two stable fixed points and one unstable fixed point

4.2.5 The Hopf oscillator with external input: a combination of hysteresis and pitchfork bifurcation

The normal form equation of the Hopf bifurcation mechanism (equation 4.3, correspond to the pitchfork bifurcation equations. As shown in the figures, the Hopf mechanism is dependent of the bifurcation parameter (μ) tuning that dictates whether the Hopf mechanism is in damping or amplification mode. Nevertheless, in the context of cochlear modelling, an additional term F representing the acoustic input is added to the equation. This variable not only changes over time but also directly impacts the value of μ. Adding this term also adds a second bifurcation mechanism: the hysteresis bifurcation. The understanding of the Hopf oscillator combined with an external signal as a combination of two bifurcation has, up to our knowledge, not been previously reported. The corresponding bifurcation equation is written as:

$$\frac{dr}{dt} = F + \mu(F)r - r^3 \tag{4.43}$$

This equation thus combines the dynamics of the pitchfork and the hysteresis bifurcation, presenting a more complex behaviour.

Intuitively, this dependence between F and μ in equation 4.43 would lead to a behaviour similar to that depicted in Figure 4.15. The system would tend to choose the solution with $r > 0$, as it would dampen high-amplitude signals and amplify low-amplitude signals, which is consistent with the expected behaviour of a cochlear model.

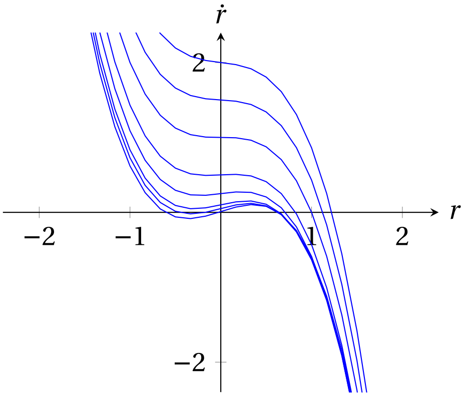


Figure 4.15: Schematic of the cochlear model bifurcation mechanism for several input signal amplitudes.

The impact of the pitchfork bifurcation is present in the signal curve, but when focusing on the solutions of the differential equation, the dominant bifurcation type is a hysteresis bifurcation. An interesting point of this graph is that we go from a large amplitude range on the y-axis to a much smaller amplitude range on the x-axis. This compression is beneficial because it allows the human ear to capture a wide range of sound amplitudes.

4.3 Simulation of oscillators

This section delves into the simulation of an array of oscillators, with the primary objective being to demonstrate the functionality of the oscillators at various stages of building an array that takes a speech signal as input. The equations to build the oscillators come from Biswas et al. (2020), who give a large overview of the oscillator capacities. We use this array of oscillator to verify its ability to capture speech input and draw some conclusions to build our own oscillator module.

First, we showcase the operation of a single oscillator, followed by the construction of an array of coupled oscillators. We then test their behaviour using both a sum of sinusoids and an actual audio signal.

4.3.1 A single oscillator

For a single oscillator experiment, we use the Hopf oscillator equation combined with an external signal $F(t)$, which is a simple sine wave.

$$\begin{aligned} z &= re^{i\omega t} && (4.44) \\ \dot{z} &= z(\mu + i\omega + \beta|z|^2) + F(t) && (4.45) \\ F(t) &= A\cos(\omega_0 t + \phi) && (4.46) \end{aligned}$$

Where:

- r : The radius of the oscillator
- ω : The inner frequency of the oscillator
- ω_0 : The frequency of the external signal
- $F(t)$: The external signal
- μ : The bifurcation parameter (fixed in these simulations
- β : The first Lyapunov coefficient, determines the stability of the system. It should be smaller than zero.

To enable the oscillator to adapt to an external signal in terms of frequency, a differential equation acting on the frequency is added.

$$\dot{\omega} = -F(t)\cos(\omega t) \tag{4.47}$$

This adaptation is illustrated in figures 4.16 and 4.17. The oscillator begins at 40Hz and adapts towards the input frequency, as shown in figure 4.17. Notably, the oscillator can adapt both to the frequency and phase of the input signal. This adaptive behaviour is crucial for the cochlear model, as it enables the OHC to adjust their response to match the basilar membrane movement.

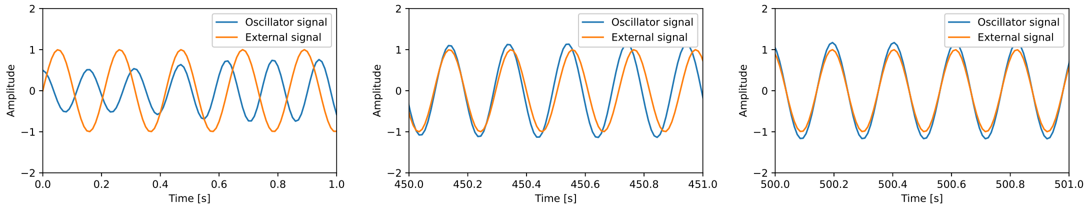


Figure 4.16: Adaptation of an oscillator to an external signal at different time steps.

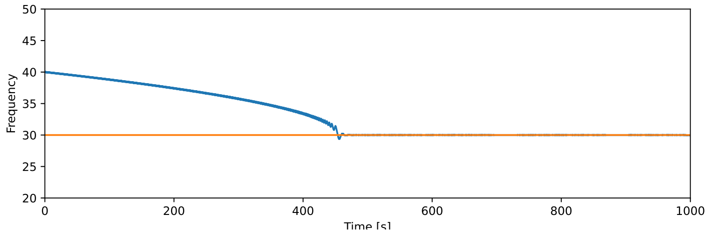


Figure 4.17: Adaptation of an oscillator's frequency ω (blue line) to the frequency of an external signal ω_0 (orange line).

4.3.2 Multiple oscillators

Oscillator equations

Since the OHCs are coupled to the other closely situated OHCs on the basilar membrane and the stereocilia are sparcely coupled through the tectorial membrane, we build an array that takes into account a certain coupling between the oscillators. Each oscillator z_i has its own Hopf bifurcation mechanism, a coupling term from the other oscillators, and an external signal (Biswas et al., 2020).

$$\dot{z}_i = \underbrace{z_i(\mu + i\omega_i + \beta|z_i|^2)}_{\text{oscillator } i} + \underbrace{\sum_{j,i\neq j}^{N} A_{ij} e^{i\frac{\theta_{ij}}{\omega_j}} z_i^{\frac{\omega_i}{\omega_j}}}_{\text{coupling with other osc.}} + \overbrace{F(t)}^{\text{ext. signal}} \tag{4.48}$$

where z_i is a complex signal that can be decomposed in $r_i e^{i\theta_i}$. The equation for each oscillator 4.48 can be separated into two functions, one expressing the radius and the other expressing the phase. This results in:

$$\dot{r}_i = r_i(\mu + \beta r_i^2) + \sum_{j,i\neq j}^{N} A_{ij} r_j^{\frac{\omega i}{\omega_j}} \cos\Big(\omega_i\Big(\frac{\theta_{ij}}{\omega_i\omega_j} + \frac{\theta_j}{\omega_j} - \frac{\theta_i}{\omega_i}\Big)\Big) + F(t)\cos(\theta_i) \tag{4.49}$$

$$\dot{\theta}_i = \omega_i + \sum_{j,i\neq j}^{N} A_{ij} \frac{r_j^{\frac{\omega i}{\omega_j}}}{r_i} \sin\Big(\omega_i\Big(\frac{\theta_{ij}}{\omega_i\omega_j} + \frac{\theta_j}{\omega_j} - \frac{\theta_i}{\omega_i}\Big)\Big) - \frac{F(t)}{r_i}\sin(\theta_i) \tag{4.50}$$

The parameters ω, ϕ and A of this equation can be made trainable. These parameters represent, respectively, the oscillator frequency, the phase difference between the different oscillators and the interaction weight between the different oscillators. The weight between the oscillators A is kept fixed, while the other parameters are adjustable.

$$\dot{\omega}_i = -\eta_\omega e(t)\sin(\theta_i) \tag{4.51}$$

$$\tau_W \dot{\theta}_{ij} = \frac{\omega_j r_i r_j^{\frac{\omega_i}{\omega_j}}}{A_{ij}} \sin\left(\omega_i\left[\frac{\theta_i}{\omega_i} - \frac{\theta_j}{\omega_j} - \frac{\theta_{ij}}{\omega_i\omega_j}\right]\right) \tag{4.52}$$

$$\dot{A}_{ij} = 0 \tag{4.53}$$

The external signal and α parameter

To enable the array of oscillators to learn the incoming signal, we introduce a retroactive system that adjusts the oscillators based on the error between the reconstructed and actual signals $e(t)$.

The difference between the incoming signal $F(t)$ and the reconstructed signal $P(t)$ is computed as the error $e(t)$. This error is then fed into the oscillators to update their parameters in an adaptive manner.

$$\begin{aligned} P(t) &= \sum_{i}^{N} r\alpha_i \cos(\theta_i) && (4.54) \\ e(t) &= D(t) - P(t) && (4.55) \end{aligned}$$

The parameter α_i learns the amplitudes at different frequencies. Its computation is based on the radius r_i and phase θ_i of each oscillator and remaining error $e(t)$. Further, a learning rate η_i controls how quick the oscillators adapt to the incoming signal. The update rule for α_i is given by:

$$\dot{\alpha}_i = \eta_\alpha e(t) r_i \sin(\theta_i) \quad (4.56)$$

The Euler step function

Hopf oscillators can be modelled using differential equations, which can be implemented in various ways depending on whether we are working with continuous or discrete signals.

For a continuous implementation, classical ordinary differential equations solvers have a significant limitation: they cannot directly incorporate an external as an argument.

For a discrete-time implementation, we can use the Euler step function, which is given by:

$$f(t_{i+1}) = f(t_i) + f'(t_i)dt \quad (4.57)$$

A manual implementation of the Euler function is a simple yet effective method that enables the integration of an external signal, such as an audio input. The Euler step shares similarities with an RNN network, commonly used in signal processing applications.

Further, more advanced methods such as the Runge-Kutta method provide a higher accuracy, but are computationally intensive. Considering the application to an ASR system, the Euler step seems like a suitable compromise between accuracy and computational requirements.

4.3.3 Experiment on an artificially built signal

A first experiment analyses how oscillators adapt to signals constructed out of a sum of sine waves. The initial experiment uses as many sine waves as there are oscillators, testing whether the oscillators can successfully recreate the signal. The second experiment uses more oscillators than necessary to reconstruct signals, analysing how oscillators interact to

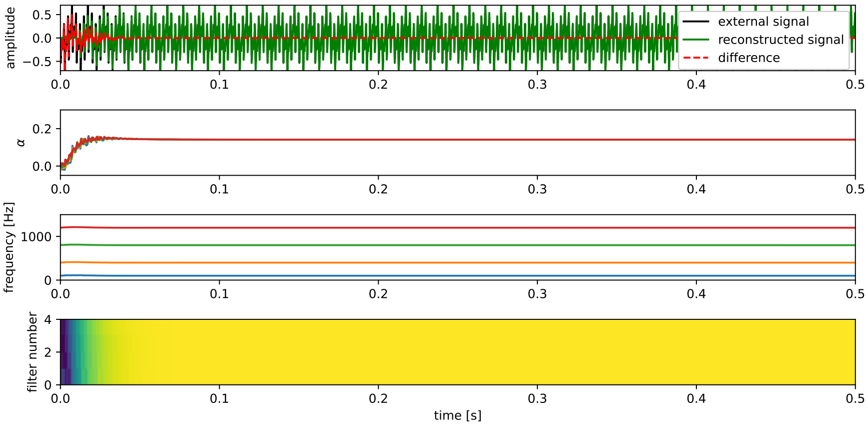


Figure 4.18: Training 4 oscillators on a signal composed out of 4 components with corresponding frequencies.

reconstruct signals when not all oscillators are needed.

Adapting N oscillators to a signal built out of N components

The first experiment consists in training four oscillators on an external built signal composed of four summed sine waves. This setup is done twice: once with the signal frequencies corresponding to the oscillator initial frequencies and again where signals are initialised to other frequencies. The goal of this experiment is to study how oscillators adapt in a presence of multiple oscillators and to compare the convergence time at various frequencies. The signal construction is given by following equation:

$$F(t) = \sum_{i}^{N} A_i \cos(\omega_i t + \phi_i) \tag{4.58}$$

$$\tag{4.59}$$

This experiment's results are presented in Figures 4.18 and 4.19. The figures are composed of the following graphs:

- A graph of the external signal $F(t)$, the reconstructed signal $P(t)$ and the difference $e(t) = F(t) - P(t)$.
- A second plot shows the α parameter adaptation, which illustrates the adjustment of the different parameters to the input signal.
- The third plot shows how the frequencies of the different oscillators adapt according to the different components of the input signal.
- The last plot shows the contribution of each oscillator in a two-dimensional space. This

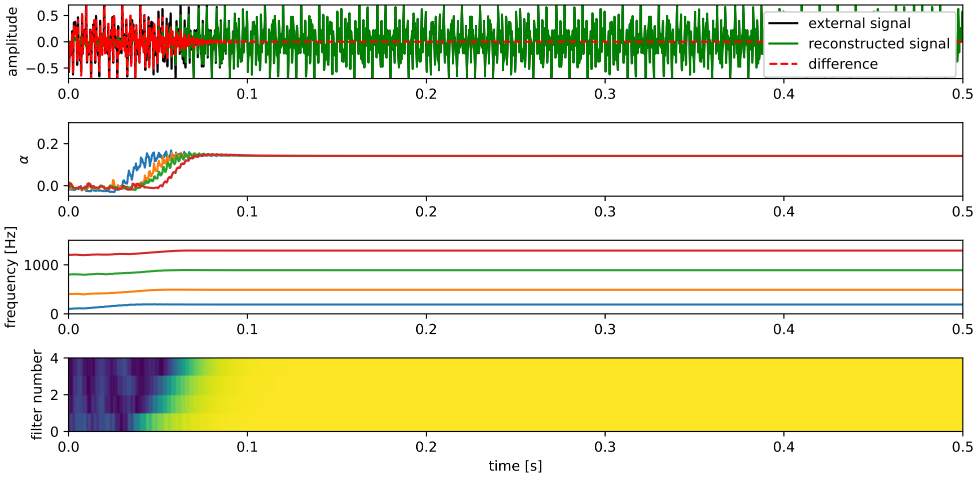


Figure 4.19: Training 4 oscillators on a signal composed out of 4 components with non-corresponding frequencies.

> representation is particularly useful when there are many oscillators, as it provides a clearer overview of which ones are actively contributing to the signal reconstruction.

It is interesting to note that the oscillators tend to adapt at the same speed to incoming signals when their frequencies match those of the signal components. In contrast when the input signal does not match the central frequency of the oscillators, for an equivalent difference in frequency, lower frequency oscillators seem to adapt more quickly than higher frequency.

Adapting M > N oscillators to a signal composed out of N components

The human ear is composed of a large array of oscillators, therefore we need an equivalent model using multiple Hopf oscillators to approach cochlear modelling. To do this, we increase the number of oscillators while keeping an input signal composed of the sum of four sine waves.

The main goal of this experiment is to analyse how an array of coupled oscillators adjust to an input signal. In a first experiment we double the amount of oscillators and in a second experiment we increase the amount of oscillators to 40.

In the case of eight oscillators (as shown in Figure 4.20), the oscillators are able to select four oscillators which are the closest to the signal frequencies and adapt accordingly. In the case of 40 oscillators (as shown in Figure 4.21), the repartition of the oscillators as it can be seen in the second plot is more chaotic. Since the oscillators are able to move, several oscillators try to adjust to the input signal. However, on the last plot, we can see four distinct lines representing

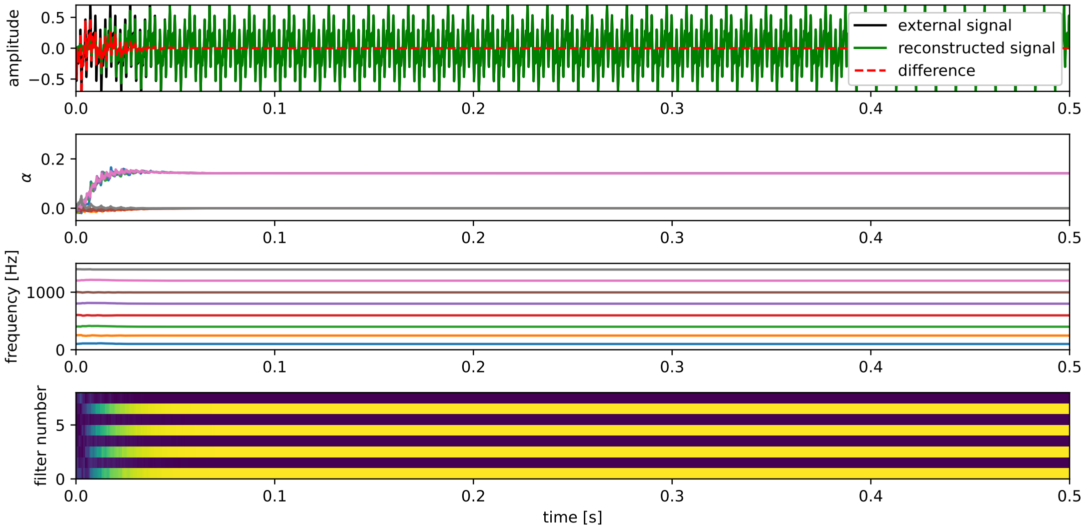


Figure 4.20: 8 oscillators and 4 components, the 4 frequencies are captured by four oscillators while the four others stand quiet.

the original sine waves that make up the input signal. Some of these lines are thicker because multiple oscillators try to adjust to the corresponding frequency component.

4.3.4 Experiment on an audio signal

The final experiment involves applying our oscillator model to an actual audio signal from the TIDIGITS dataset introduced in section 2.3.4. We select a short audio snippet from this dataset, which contains recordings of spoken digits. For this experiment, we use 100 oscillators arranged in a mel-spaced array[I].

Figure 4.22 shows the adaptation of the oscillators to the audio signal. The reconstructed signal (on the first plot), shows that the oscillator output doesn't perfectly match the original input waveform. This is because our model needs time to adapt to the evolving audio signal as it continues to play back in real-time. Although some of the oscillators are triggered correctly, there's a noticeable delay in their activation, which makes it challenging for them to follow changes in the speech input. In the second and fourth plot, we see that different oscillators are activated alternatingly. The selected oscillators correspond to the audio frequencies, but are not able to keep up with slight changes. Nevertheless, when comparing the oscillator activation and the mel spectrogram, we clearly see that the oscillators try to adapt to the incoming signal around the right frequencies.

[I] The number of oscillators and their distribution has been chosen arbitrarily for this experiment, in Chapters 5 and 6 we further analyse the number of filters and distribution a system tends to learn when trained with a gradient descent algorithm.

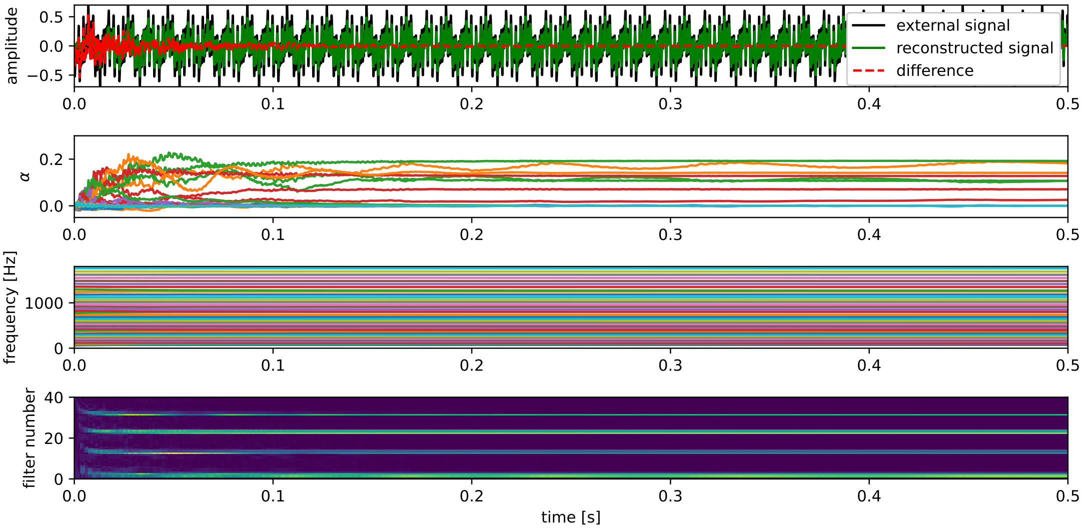


Figure 4.21: 40 oscillators and 4 components, the 4 frequencies are clearly appearing on the image representing the different sine wave of the input signal.

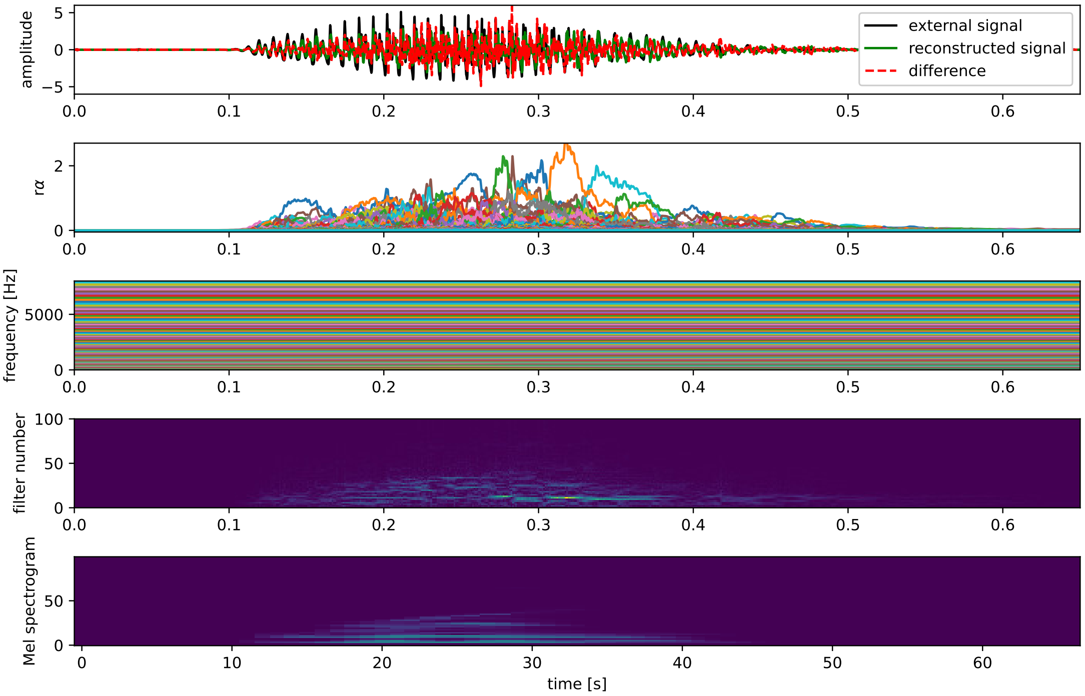


Figure 4.22: 100 oscillators capturing the information of an audio signal.

4.3.5 Limitations of the simulation model

Biswas et al. (2020) propose an implementation of the Hopf oscillator model with an adaptive frequency and coupling factor for the oscillators. The different experiments demonstrate that Hopf oscillators are able to capture the input signal information, which was the main goal of these experiments.

However, this initial simulation model has its limitations. Consisting of four nonlinear differential equations, it is computationally demanding to integrate into a trainable ASR system. On the one hand, having a large number of oscillators with a trainable frequency provides interesting insights of how oscillators are adapt and move in response to the input signal. Analysing those movements into an ASR context could reveal valuable information about the optimal oscillator distribution. On the other hand, the frequency adaptation mechanism has some limitations. Oscillators that change frequency do not return to their initial positions, which means they can switch positions over time, which is physiologically not plausible. Additionally, introducing another differential equation for frequency adaptation increases computational demands in a deep learning training context.

For the further research we propose two main direction:

- **Simplify the cochlear model** : We plan to use a simple cochlear model with trainable frequency and bandwidth to identify the key trends that filters would learn within an ASR system while maintaining computational realism.
- **Adapt the Hopf oscillator model from section 4.3.2**: Building on the insights gained from the simplified cochlear model, we aim to design a Hopf oscillator module with fixed central frequencies, using the properties learned by trainable filters to determine the optimal oscillator distribution.

The coupling mechanism in our model effectively compensates for input signal variations through interactions between oscillators. However, this mechanism is computationally heavy and time-consuming to execute. To address this challenge in an ASR module context, we decide to remove the explicit coupling mechanism and instead use the output of individual oscillators as input to the ASR system through backpropagation.

4.4 Precomputed plausible cochlear features in an ASR system

An interesting avenue of research is analysing the performance of physiologically plausible cochlear features in a classical ASR system. For this analysis, we utilize the CARFAC model proposed by Lyon (2017b) to generate features and compare their ASR performance to that of MFCC features. CARFAC has the main advantage to contain the different structures of the organ of Corti in its model. Moreover, it integrates an active gain control inspired from the active amplification mechanism present in the cochlea. Therefore, CARFAC is the closest model to the Hopf oscillators which can deliver speech features that can be used for ASR assessment.

CARFAC features are similar to traditional features commonly used in ASR, with the difference that they are generated by a plausible cochlear model. Evaluating these features in a classical ASR system provides insight into how well physiologically plausible features integrate with such systems. To ensure comparability with standard ASR features, the CARFAC features used as input to the ASR system are computed as follows:

$$y[n] = \log\left[\sum_{i=0}^{L}(x[n\cdot w + i]\cdot h[i])^2\right] \tag{4.60}$$

Where x represents the input signal, L is the length of a 25 ms window, w the 10 ms time shift for the computation between the different frames and h is a hanning window.

Table 4.1 provides a comparison of the ASR performance between MFCC and CARFAC features on the TIMIT dataset (Garofolo, 1993). On an MLP network both with and without a context window and on a Light Gated Recurrent Unit (Li-GRU) which is a recurrent neural network.

	MFCC	CARFAC
MLP without context window	21.4	23.9
MLP with context window	18.2	20.2
Li-GRU	15.6	16.5

Table 4.1: Phone error rate comparison between MFCC and CARFAC features.

Overall, MFCC features outperform CARFAC features across all configurations. This suggests that classical ASR systems are less suited to features incorporating cochlear characteristics compared to MFCC features. However, given the relatively close performance, CARFAC features still show their capability to effectively handle ASR tasks.

4.5 Conclusion

The current state-of-the-art understanding of the cochlea is represented by the Hopf oscillator model, which has been shown to capture some key characteristics of the active amplification mechanism. The mathematical understanding provided by this model is considered the most advanced in the literature.

In this chapter, we demonstrated using a simulation model based on Biswas et al. (2020), the Hopf oscillator model can indeed process audio signals and the activation pattern approximates the mel spectrogram of the same audio snippet. Nevertheless, there are also limitations to this approach: the model relies on a series of non-linear differential equations, which makes recurrent implementation computationally demanding.

Moreover, our simple cochlear feature experiment revealed that using more physiologically plausible features for ASR does not necessarily lead to better performance compared to classical MFCC features. Studying plausible physiological models gives some interesting constraints to the core ASR model to compute more physiologically realistic speech recognition capacities, but will, a priori, not provide performance or computationally competitive solutions.

The research in this thesis is further built in two steps:

- Use a filter implementation with trainable filters to derive the main characteristics a filterbank would learn to determine the amount of filters and the filter distribution that are induced within an ASR system. Therefore we use both a simple ASR system in chapter 5 and a self-supervised model in chapter 6.
- Develop a Hopf oscillator module that can be integrated into an ASR system. For computational reasons, this module will have fixed central frequencies and the coupling term will be omitted, knowing that further processing in the ASR system is able to recombine the different oscillator outputs. This work is presented in chapter 7.

5 Trainable filters

Previous chapters 3 and 4 introduced the concept of modularity and the understanding of the Hopf oscillator as a mathematical approach to model the cochlea as illustrated in Figure 5.1. In this chapter, we focus on a first approach that combines a simple ASR system with a simple cochlear model: a trainable filterbank as part of a downsampling CNN. A filterbank is a first approximation of a cochlear model and is used for feature extraction in ASR mechanisms.

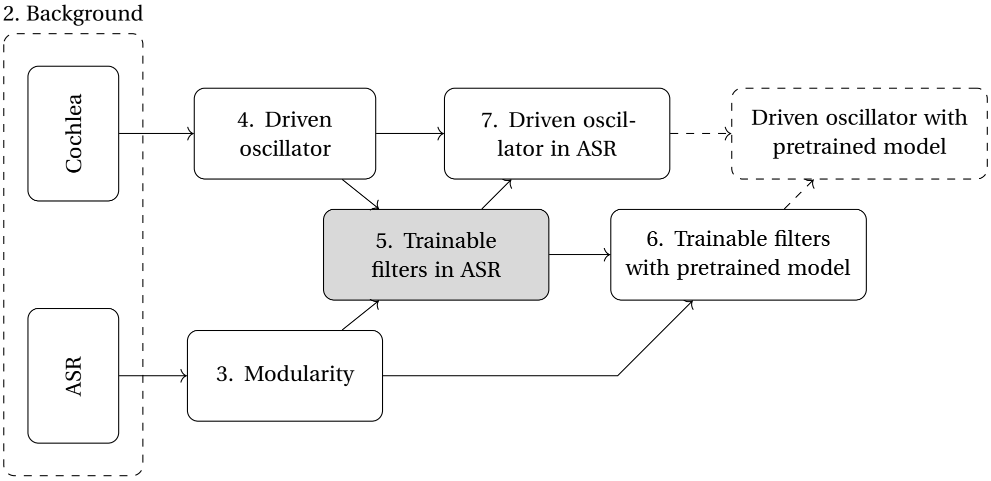

2. Background
Cochlea
ASR
4. Driven oscillator
7. Driven oscillator in ASR
Driven oscillator with pretrained model
5. Trainable filters in ASR
6. Trainable filters with pretrained model
3. Modularity

Figure 5.1: General overview of the thesis.

Current ASR systems perform very well from a machine learning perspective, this suggests they are emulating the human hearing system effectively. In this chapter, we are investigating how an interpretable filter layer evolves in the context of an ASR training mechanism and analyse the filter distribution with different constraints. Whilst neither the ASR nor the trainable filterbank are new, the main goals of this chapter are the following:

- **Creating a baseline ASR structure** that works with a trainable filterbank. This baseline structure can serve as a starting point for integrating more complex cochlear models, allowing individual modules to be replaced while maintaining overall network functionality.

- **Defining some hyperparameters** such as the filter distribution and the number of filters required for ASR processing. For time and computational purposes, we define these hyperparameters by analyzing what trainable filterbanks tend to learn in an ASR context.

Besides, we demonstrate results on interpretability of the resulting filters that were not reported by the original authors and provide signal processing adjustments that improve the overall performance. The majority of the text in this chapter was originally published as:

> Coppieters de Gibson, L., & Garner, P. N. (2022). Low-level physiological implications of end-to-end learning of speech recognition. *Interspeech 2022*, pages 749–753. doi:/10.21437/Interspeech.2022-10093

5.1 Introduction

Advances in automatic speech recognition (ASR) have led to performance that is very good in terms of WER, but perhaps at the expense of our own understanding of how they function. End-to-end (E2E) techniques (Amodei et al., 2016) have removed the need for knowledge of the hearing mechanism. Self-supervised training (Schneider et al., 2019) has done the same for phonetics. More generally, large pre-trained models are available (Babu et al., 2021) removing the need for even the ML know-how. Given that these systems work well, the question arises: "what have they learned?" This is difficult to answer because their component parts cannot readily be mapped to biological ones.

In this study, we are interested in the hearing mechanism. The biological mechanism is quite well understood (Lyon, 2017a), with important parts being the logarithmic response to both frequency and amplitude. For many years, filterbank approaches were used as models of this process (Hermansky, 1990a; Juang & Rabiner, 2005). Whilst many variations have been studied, the authors' ad-hoc experience suggests that the details do not lead to big changes. Recent E2E approaches, however, have clearly demonstrated that training the filterbanks can be beneficial (Collobert et al., 2016). A (1D) convolution layer in the machine learning field is a filter in the signal processing field. However, the only validations of which we are aware tend to show that the component convolutions learn filters with a distribution similar to a mel filterbank. This in turn tends to reinforce the above question rather than answer it.

In SincNet , Ravanelli et al. (Ravanelli & Bengio, 2018b) constrained the convolutions to be a sinc ($\sin(x)/x$) form, leading to a rectangular band-pass filter. The filter is then defined by two trainable parameters: the lower frequency and the bandwidth. Whilst not being biologically accurate, this approach has a distinct attraction of being interpretable.

In the remainder of this mainly experimental paper, we describe SincNet and a modest frame-based experimental scenario. We confirm that SincNet learns a mel filterbank, but also show that wider bandwidth filters are important for performance. We argue that such filters arise because of restrictions of standard ML convolutional architectures, and conclude with what this infers about how to construct a biologically plausible hearing model.

5.2 Background

The study of the human cochlea has interested many researchers since the beginning of the 20th century. Von Békésy laid the groundwork of the research on this topic in 1960 (Von Békésy, 1960).

The basilar membrane in the cochlea can be interpreted as a natural filterbank (Geisler, 1976; Zwislocki, 1953). Current understanding of the working of the cochlea is that wave propagation is an active process (De Boer, 1983) and works as an array of Hopf oscillators (Hudspeth et al., 2010a; Hudspeth, 2008). However, in this study, we limit ourselves to passive analogues. The scaling of this natural filterbank has been analysed from different points of view, leading to several scaling (or warping, spacing) approaches. The Greenwood scaling (Sridhar et al., 2006) is the one that best represents the scaling of the frequency sensitive hair cells in the cochlea based on the physical distance on the basilar membrane of the hair cells. The mel scale (Stevens et al., 1937) is based on frequencies judged to be equally spaced in human perceptual tests. Bark (J. O. Smith & Abel, 1999; Zwicker, 1961) and ERB (equivalent rectangular bandwidth) (Moore & Glasberg, 1983) are somewhere between mel and Greenwood, but by contrast are derived from critical *bands* of hearing.

ASR frontends take either some preprocessed features as input or, more recently, raw input waveforms. Filterbanks have been the basis of feature extraction (Shannon & Paliwal, 2003) for many years. As early as 2001, a study (Burget & Heřmanskỳ, 2001) showed that a filterbank could be obtained from a mathematical derivation of a data driven design. From the resulting filterbank, about half of the filters were then kept to represent the filterbank motivated by the fact that those filters were enough to cover the whole frequency range. For the E2E approaches, CNNs for ASR were introduced by Hinton et al. (Hinton et al., 2012; Palaz et al., 2013b) and have been used for a decade. Since 2018 some architectures propose a way to combine both the filterbanks and the E2E architecture, where the filterbanks are trainable and part of the convolution layers. Zeghidour et al. (Zeghidour et al., 2018) proposed an implementation with with Gabor filters and Ravanelli et al. (Ravanelli & Bengio, 2018c) with rectangular filters (SincNet). Other work on trainable filterbanks includes that of Seki et al. (Seki et al., 2019), who proposed an architecture based on a filter layer combined with a deep neural network (DNN) where the filter features were directly computed with a log-compression after the filter layer. In that study the gain, central frequency, bandwidth and filter shape were free to train, whilst in SincNet only two parameters are free to train, defining the filters in the first layer.

5.3 Initial Analysis

In an initial, quite basic analysis, the main motivation was to understand what the trainable filters learn; i.e., which typical hyperparameters (e.g., the number of filters needed to describe the signal) can be derived from those trainable filter models, which initializations are appropriate. For this study, we focus on SincNet.

5.3.1 SincNet setup

The SincNet model (Ravanelli & Bengio, 2018c) is built with a 4-layer CNN followed by a 5-layer DNN. The first layer of the CNN is constructed with trainable filters. Those filters are initialised as a rectangular bandwidth mel-scale filterbank, an easily computable type of filter in the time domain. Since the inverse Fourier transform of a rectangular low-pass filter is a sinc function, a rectangular bandwidth filter can be written as the difference of two low-pass filters as in equation 5.1.

$$h[n] = \mathrm{sinc}(2\pi f_2 n) - \mathrm{sinc}(2\pi f_1 n), \tag{5.1}$$

where $h[n]$ represents a typical filter of the first convolutional layer. The trainable parameters of the SincNet filters are the lower frequency (f_1) and the bandwidth ($f_2 - f_1$), i.e., linear combinations of f_1 and f_2. Moreover, in the time domain, filtering a signal is mathematically equivalent to the convolution of this signal with the filter kernel.

Between the different convolution layers the following operations are used: maxpooling for downsampling, layernorm, ReLU and dropout before passing through a five-layer DNN. The input of signal is a raw waveform of 200 ms at 16 kHz. For this research the experiments are performed on TIMIT (Garofolo, 1993) and to verify that the observations are not database related, the baseline experiments have been double checked with the mini-Librispeech database (Panayotov et al., 2015).

5.3.2 Method

The experiments in this chapter are conducted using the pytorch-kaldi framework (Ravanelli et al., 2019). For the baseline experiment, we use the existing SincNet configuration with 128 filters in the Sinc layer, followed by three convolutional layers. The kernel size is set to 129 for the Sinc filters, and 5, 5, and 3 for the standard convolutional layers. Max-pooling is applied after each layer with downsampling sizes of 3, 3, 3, and 2, respectively. We use ReLU as the activation function and layer normalization for regularization.

The SincNet network takes raw speech as input, with utterances segmented into chunks of 200 ms. The system outputs a probability vector over phoneme classes for each frame at the encoder level. These predictions are compared to the target labels, and the error between the predicted and actual labels is used to compute the loss. This loss is then backpropagated through the network to update the weights of the different layers.

The experiments are conducted on the TIMIT dataset (Garofolo, 1993), which is described in more detail in Section 2.3.4. The training is fixed to 24 epochs, the different results are reported on a single run.

5.3.3 Baseline

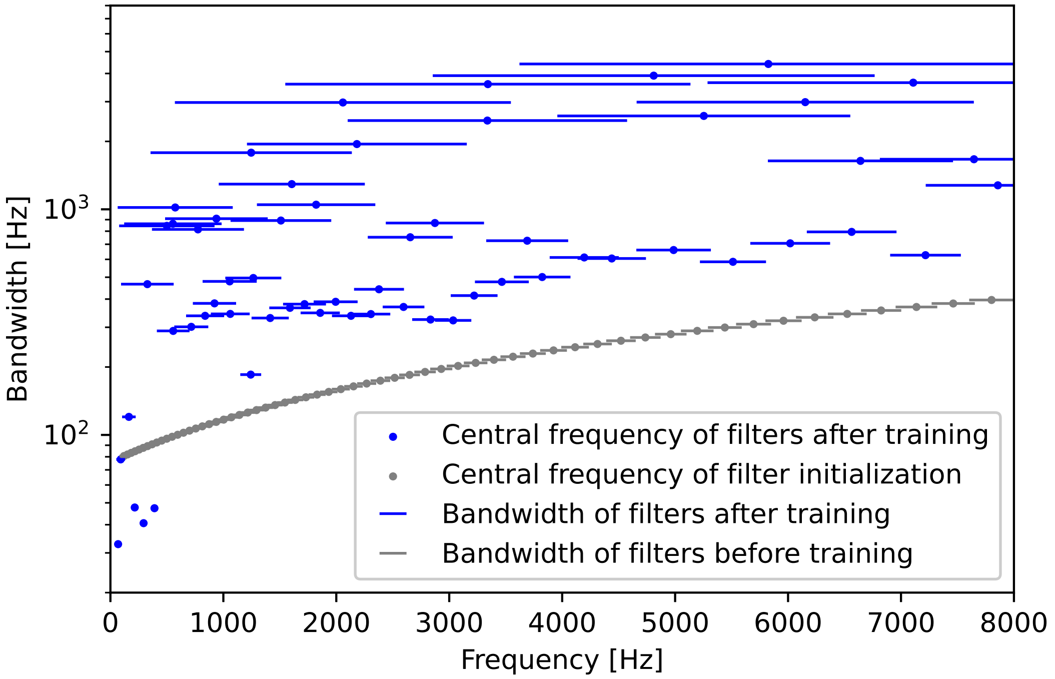


Figure 5.2: Evolution of the baseline implementation of SincNet: the grey graph shows the initial filter distribution and the blue graph shows the filter distribution after training. The x-axis represents the frequency range and the y-axis the amplitude of the filters. The filters themselves are represented by their central frequency (dot) and their bandwidth (bar).

In the default implementation, the number of filters is initialised to 128 followed by 3 CNN layers of 60 filters each. The filter distribution for a similar experiment (60 filters on the first layer) is illustrated on figure 5.2. We observe that some filters with a comparatively narrow bandwidth train towards a filterbank. The others train towards wider bandwidth filters. Concerning the frequency range, the wide-band filters could in principle be reconstructed with a linear combination of narrow band filters. In this paper, those two types of filters will be refered to as narrow and wide-band filters. The first part of this study focuses on the narrow band filters, since a large number of the wide-band filters seem to overfit the data.

Table 5.1: Filter pruning experiment: numbers of narrow band filters and related PER in function of the initialization.

Sinc-Layer num. filters	CNN-layers			narrow band filters	PER (%)
128	60	60	60	39	17.1
100	60	60	60	45	17.1
80	60	60	60	38	17.2
60	60	60	60	32	17.4
40	60	60	60	27	17.5
30	60	60	60	24	17.5

Table 5.2: SincNet experiment: compare the performance of the training with the filters fixed and the filters that are free to train.

	fixed filters		trained filters	
num. filters	loss	PER	loss	PER
40	2.35	18.3	2.31	17.6
30	2.37	18.0	2.33	17.5

.

5.3.4 Number of filters

Some filters in the first convolutional layer stay narrow-band while the others train towards wider bandwidths. Table 5.1 gives an overview of the number of narrow band filters as well as the PER on TIMIT. To determine the number of narrow band filters an ad-hoc pruning operation has been applied after the filter training: the filters with wide bandwidths covering parts of the spectrum that are already covered by smaller bandwidth filters are discarded and only one filter is taken into account around the Nyquist frequency.

The number of filters needed by the model to build a filterbank covering the whole frequency range can be determined by the number of narrow band filters. From table 5.1 we can deduce that above 30 filters, the number of narrow-band filters that describe the frequency range is around 30 - 40 filters, this correlates with the results obtained by Zeghidour (Zeghidour et al., 2018) using Gabor filters.

We also notice that when the first layer is initialized to 30 or 40 filters (corresponding to the number of narrow-band filters of other layers), some of those filters still train toward larger band filters. To analyse whether keeping the initilization to the initial scale performs as well as the combination of narrow and wide-band filters that the model learns, experiments have been made on 30 and 40 filters for fixed and non-fixed filters, the results are given in table 5.2. This raises the hypothesis that the wide-band filters are bringing some information not provided by the narrow band filters.

Table 5.3: Mean Euclidean distance between narrow bandfilter's normalized central frequency distribution and different scalings for different amount of filters (Mel filterbank) and different initial scalings (30 filters).

Initialized to scale - filters	Compared to Mel	Bark	ERB	Greenwood
Mel - 128	**0.0023**	0.0047	0.0070	0.0086
Mel - 60	**0.0018**	0.0044	0.0070	0.0088
Mel - 40	**0.0022**	0.0039	0.0065	0.0084
Mel - 30	**0.0020**	0.0043	0.0071	0.0091
Bark - 30	**0.0025**	0.0037	0.0062	0.0082
ERB - 30	0.0030	**0.0029**	0.0055	0.0076
Greenwood -30	**0.0037**	0.0068	0.0095	0.0116

5.3.5 Scale after training

Given that there are several frequency warpings that can be justified from a physiological point of view, it is informative (and simple) to investigate which one is preferred by an E2E system. It is clear by inspection that it is the narrow band filters that learn the warped filterbank. In (Agrawal & Ganapathy, 2019) a convolutional variational autoencoder (CVAE) architecture that learns convolutional filters from raw waveforms using unsupervised learning was proposed. However, the analysis was only based on the central frequencies learned by those filters, not the narrow/wide-band distinction. In the present paper the central frequencies of only the narrow band filters are taken into account.

Experiment

The experiment consisted simply of analysing which filterbank the narrow-band filters trained above were learning. The experiment was repeated for several models with different initializations. The metric used to compute the distance between the initial and trained scale is the mean of the Euclidean distance:

$$d(x,s) = \frac{1}{N}\sqrt{\sum_{i=0}^{N}(x_i - s_i)^2} \tag{5.2}$$

The narrow band filters of a filterbank initialized to the mel scale remain mel-scale distributed. When initializing 30 – 40 filters as starting point to different scalings, other scalings also train towards mel scale. It follows that the mel scaling is an appropriate choice for filterbank initializations.

5.3.6 Corollary

The results so far have reinforced that E2E approaches do indeed learn what has been known for many years about cochlear models: that 30 to 40 filters are sufficient and that, regardless of physical measurements, the mel scale is the one that is perceptually important. However, from figure 5.2 it is clear that SincNet filters train towards a mixture of narrow *and wide-band* filters. Moreover from table 5.1, in all the experiments done for this section, the model always learns wide-band filters. It follows that these wide-band structures are important. Two questions arise:

1. Can the filters be initialized to wide-band, removing or reducing the need to train them?
2. Why do wide-band filters appear at all given that they are, to a first approximation, just linear combinations of narrow-band filters?

These are addressed in the following section.

5.4 Wide-band filter analysis

5.4.1 Wide-band initialization

Hypothesis

Wide-band filters are important; it follows that the training can be optimized by initializing a narrow band filterbank as before and adding wide-band filters in addition of those filters. This hypothesis can be confirmed by an experiment comprising initializing several frozen superimposed filterbanks where the wide-band filters are combinations of several narrow band filters.

Experiment

Figure 5.3 shows the initialization and training of a model built with 4 ranges of filters illustrated on the upper plot.

An estimation of the filter distribution after training of this new initialisation is illustrated in the bottom plot. Four experiments using those filters are summarized in table 5.4:

Using only the narrow band filters gives a final PER of 18%, the combination of narrow and wide-band filters give for frozen filters a PER of 17.7% and for trained filters a PER of 17.5%. A combination of narrow and wide-band frozen filters already gives an improvement of 0.3% PER. The same effect is observed on the loss: the loss for a combination of narrow and wide-band filters is lower than for only narrow band filters. The new initialization is consequently closer to what the model learns compared to the baseline initialization. Aside, it is interesting

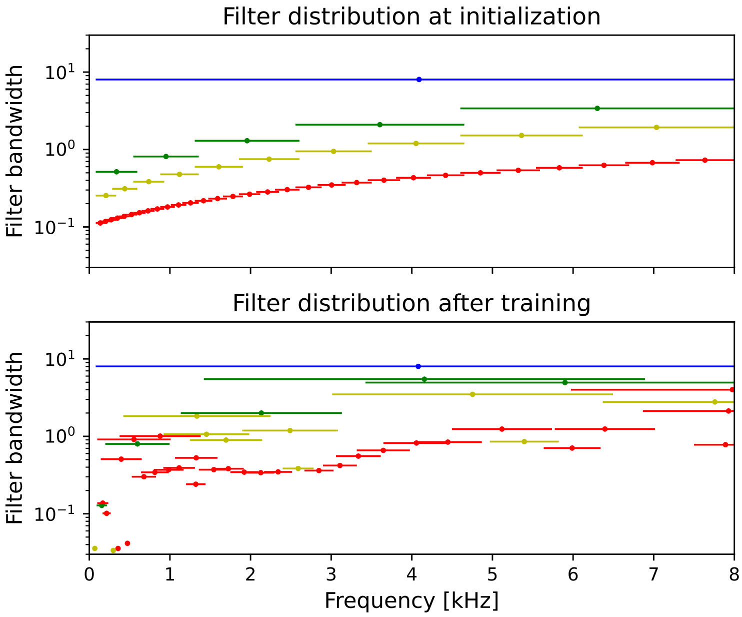


Figure 5.3: Filter repartition of superimposed filterbanks before (top plot) and after (bottom plot) training. In the initialization, the red filterbank is a narrow-band filterbank composed of 30 filters. The rest are filterbanks of 10, 5 and 1 filters capturing information that could in principle be reconstructed by a combination of the narrow band filters.

Table 5.4: Summary of experiments using narrow and/or wide-band filters

filters	filter type	fixed	loss	PER
10 - 5 - 1	wide	yes	2.410	18.6%
30	narrow	yes	2.374	18.0%
30 - 10 - 5 - 1	narrow & wide	yes	2.335	17.7%
30 - 10 - 5 - 1	narrow & wide	no	2.306	**17.5**%

to notice from figure 5.3 that when trained most of the narrow band filters stay narrow band and most of the wide-band filters stay wide-band.

Thus the hypothesis is demonstrated. We conclude that it can indeed be beneficial to provide a mixture of narrow and wide-band filters in an ASR front-end.

5.4.2 Why wide-band filters?

Wide-band filters are in principle linear combinations of several narrow band filters; the network should be able to learn such a combination trivially, much as we assume it is learning the energy feature that was commonly used in HMM-based models. The most plausible explanation for the network's failure to do so arises from the interaction of harmonic (voiced) and aperiodic (unvoiced) components. Harmonic components in the same filter add constructively in the magnitude domain. Aperiodic components, however, add as random variables; the variances add leading to a magnitude reduction by a factor of $\sqrt{N}$ for N discrete components. The wider band filters hence tend to favour the voiced components.

In complex narrowband (e.g., Fourier transform based) filters, the squaring operation leads to both harmonic and aperiodic components adding *in the same ratio* in the magnitude or power domain, inhibiting emphasis of the harmonic component. SincNet comprises real-valued filters; however, the subsequent network architecture can inhibit the behaviour. Each convolutional layer is followed by four typical operations: max-pooling (downsampling), layernorm, ReLU (activation function) and dropout that have some influence on the signal. Of these, the maxpooling function and ReLU activation bring some distortions to the signal.

Hypothesis

It is possible to design a simple experiment to examine whether the above non-linearities inhibit simulation of wideband filters. The experiment encompasses three intuitive hypotheses:

1. **Using average-pooling instead of maxpooling** removes the noise artifacts that are created by maxpooling on the filtered signal, but since we continue to use a pooling function, aliasing still happens for the high frequencies. In (Dubey et al., 2019) some experiments showed that using average pooling reduced the PER but without explaining the possible reasons.
2. **Moving the first downsampling factor towards a further layer** inhibits downsampling just after the filtering of the signal, this removes both the effect of aliasing and signal distortion (although it increases the data size at several convolution layers).
3. By inspection, **using a tanh or sigmoid function** removes some low frequency artifacts created by the ReLU function. However, it is well known that the cochlea contains a rectifier function (De Boer & De Jongh, 1978), implying that ReLU is the right physiologically plausible solution. It is not clear a-priori which of these properties is more important.

Experiment

Table 5.5 summarises the performance of the experiments implied above after the first convolution layer: use average pooling, move the first downsampling factor to a later layer and check that ReLU is appropriate.

Replacing the max-pooling with average-pooling leads to an improvement in performance. Displacing the downsampling by one layer, in principle allowing the network to combine filters, leads to a further improvement. This broadly demonstrates the first two points of our hypothesis. Changing the activation function to sigmoid deteriorates the PER. This tends to confirm that the physiologically-implied rectifier — yielding a simple spectral envelope — is also the right solution in the artificial solution. We note that, even with the best performing architecture, the system still learns some wide-band filters. This implies that our solution is

Table 5.5: The effects of modifying the downsampling and pooling schemes. The numbers in the second column refer to the downsampling rate at each of the pooling operations in the convolutional layers (1 implies no downsampling).

filters	downsampling	pooling 1st layer	act. function	PER
30	3-3-3-2	max	ReLU	17.5%
30	3-3-3-2	avg	ReLU	17.1%
30	1-3-3-6	-	ReLU	**16.8**%
30	1-3-3-6	-	sigmoid	18.1%

not perfect.

5.5 Conclusion

E2E training of filterbanks for ASR leads to filters that resemble a standard filterbank. However, wider bandwidth filters are learned too, and are important for good ASR performance. The central frequencies of the narrow-band filters tend to a mel spacing, regardless of the initialisation. This confirms a well understood mechanism, suggesting that it exists in the biological system. There appears to be an optimal number of filters — around 30 to 40 — that also correlates with acknowledged literature.

We suggest that wide-band filters are learned to distinguish voiced (harmonic, coherent) components from either background noise or unvoiced (aperiodic, stochastic) components. In principle, a network should be able to learn such wide-band components by combining narrow-band ones. We argue that this capability is precluded by the (otherwise standard) ML architecture; in particular the phase will be lost by maxpooling. This argument is borne out by experiments showing that a structure retaining a more formal down-sampling mechanism can lead to better performance.

We are not aware of structures in the inner-ear or cochlea that can emulate physical wide-band filters. However the phase information is retained in our current best understanding of cochlear operation, retaining also the possibility that such filters are emulated in the auditory path. Proving this would be difficult, perhaps requiring some combination of selective stimulii with MRI or EEG sensing. It remains as a hypothesis for the neuroscience community.

Our own future work will focus on more biologically plausible architectures for the cochlea. This experimental study indicates that any such model will need to retain phase in order for the subsequent network to take advantage of both narrow-band and wide-band features.

6 Trainable filters with self-supervised pretrained model

Auditory research aims generally to lead to an understanding of physiological processes. By contrast, the state of the art in ASR (notably recognition) is dominated by large pre-trained models that are meant to be used as black-boxes. In this chapter we aim to combine state-of-the-art ASR with a trainable filter model, which have a main interpretability advantage above classical CNN structures.

As depicted in Figure 6.1, this chapter further builds on Chapter 5, in which we presented a first study of trainable filters in a simple ASR structure and we analyzed the behaviour of SincNet filters in a simple ASR network. Further, we use the modularity concept introduced in chapter 3 to combine the trainable filterbank approach with a pre-trained transformer-based ASR network .

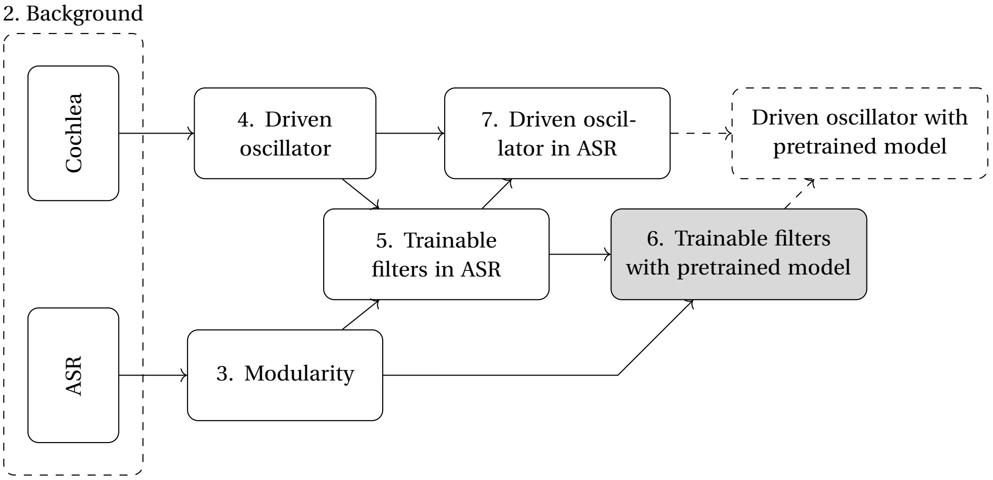

2. Background
Cochlea
ASR
4. Driven oscillator
7. Driven oscillator in ASR
Driven oscillator with pretrained model
5. Trainable filters in ASR
6. Trainable filters with pretrained model
3. Modularity

Figure 6.1: General overview of the thesis.

The research in chapter 5, showed two types of filters appearing: narrow-band filters and wide-band filters. For narrow-band filters, we identified those that more or less followed the conventional expected frequency bands and covered the whole frequency range. These narrow-band filters showed characteristics of a typical ASR filterbank: it was composed of 30 to 40 filters, and the scaling of the central frequencies tended to learn is the mel scale. By contrast, the wide-band filters were filters that learned more unexpected wide-band structures. The reason for this kind of structure was not clear, especially given that, theoretically, wide-band structures could be reconstructed by linear combinations of the other filters.

The overall goals of this work are:

- Analyze whether the wide-band structures still appear when SincNet filters are combined with a large self-supervised network.
- Analyze the number of filters that try to capture the network content.
- Compare the performance of this interpretable approach with the classic CNN structure in a self-supervised context.

This work also represents a main implementation challenge. Self-supervised networks are usually combined with a final layer that learns to combine latent space components into labels (words, phonemes, speakers, or other task-related labels). This can easily be done by allowing the gradients to backpropagate only through the backend layer without impacting the weights of the transformers. For longer fine-tuning, once this backend layer has been trained, the whole model can train together.

In this work, we show that the hybrid system can be trained and evaluated with various combinations of fine-tuning and self-supervision. To introduce an interpretable frontend, we replaced the existing frontend layer, which implies backpropagating through the whole network to be trained. Using separate optimizers for the different modules, we propose an approach that is capable of first training the frontend and backend modules and then fine-tuning the whole ASR system.

We conclude that using a hybrid structure is an appropriate way to proceed in auditory research, more generally allowing the work to take advantage of larger state-of-the-art models and databases from which it would not otherwise benefit.

The content of this chapter was originally published in the following work. However, some background material has been omitted to prevent duplication within the thesis.

> Coppieters de Gibson, Louise, and Philip N. Garner. "Training a Filter-Based Model of the Cochlea in the Context of Pre-Trained Acoustic Models." *Acoustics. Vol. 6. No. 2. MDPI, 2024.*

6.1 Introduction

Since the advent of deep learning, the general field of speech technology has advanced to a point that would be unrecognizable to proponents working just a decade ago (Hinton et al., 2012; Seide et al., 2011b). Perhaps the main difference is that the technology is now driven by the machine learning community rather than the speech processing community. One visible effect of this change is that physiologically inspired approaches that guided the topology of hidden Markov models (HMMs, the previous state of the art) have largely disappeared; they have been replaced by "end-to-end" (E2E) approaches. These in turn learn the representation that leads to the best performance on the available data, irrespective of physiological plausibility. For example, HMMs were typically defined at a phoneme level, but deep architectures have preferred byte-pair encodings (Sennrich et al., 2015) or often just (orthographic) letters. Spectral features have been replaced by generic outputs from convolutional layers. The exemplar in these cases is perhaps the "wav2letter" of Collobert et al. (Collobert et al., 2016).

Self-supervision in particular has been a key recent advance in deep learning, in computer vision as well as audio processing (Schneider et al., 2019). The "self" supervision arises from a loss-function that is designed to reveal discriminability at a given granularity of interest. Crucially, this removes the need for labeled training data, resulting in systems that can be trained on vast amounts of data. To put this into perspective, the LibriSpeech database of . (Panayotov et al., 2015) is one of the largest commonly available labeled speech databases at around 1000 h. By contrast, self supervision is associated with datasets of tens of thousands of hours; the million hours of (Parthasarathi & Strom, 2019) is roughly a century. The commercial side of the community has made pre-trained models available (Babu et al., 2021; Ott et al., 2019). Note that this pre-training implies almost fully trained; it is distinct from the pre-training that used to be applied to some networks to enable them to respond to conventional training (G. Dahl et al., 2012). However, subsequent training of the recent models is common and referred to as fine-tuning.

Given that these recent systems work well, the question arises: "what have they learned?". This is difficult to answer because their component parts cannot readily be mapped to biological ones. A recent study has shown that analogies can be drawn between layers of such systems and brain function (Millet et al., 2022), suggesting that the two fields are in fact converging. In general, however, it seems reasonable that in order to make inferences about biological function, it is necessary to build the deep networks using components that are themselves interpretable. Until quite recently, embedding such components into a deep-learning framework might have been onerous owing to the need for their being differentiable.

Fortunately, current automatic gradient packages allow networks of arbitrary operations. They also allow arbitrary parts of a network to be trained. This automatic gradient method is a technique that automates the calculation of gradients for the model parameters, enabling stochastic gradient descent computation. It alleviates the need for manual derivation while enhancing the efficiency and scalability of these gradient-based algorithms.

The innovation of this work lies in the transformative impact that a self-supervision model brings to the auditory model. Therefore, we combine a current state-of-the-art pre-trained model with a trainable filter front-end to infer a physiologically plausible function of the human auditory apparatus. More specifically, we build upon a previous study (Coppieters de Gibson & Garner, 2022) where we showed that attempting to do this with a smaller model led to unexpected results. We show that it is possible to use a state-of-the-art model, and that doing so mitigates the effects of the smaller model. An interesting related work also analyses the evolution of a filter-based initialization of a filterbank (Vieting et al., 2023). The main difference from our approach is that we use SincNet filters, which maintain a rectangular shape, whereas they adopt a classical CNN-based approach.

Section 6.2 first describes the state of the art in both E2E modelling as well as cochlear modelling for speech technology. Section 6.3 further details how a simple auditory model can be trained from scratch in the context of a larger model that has been pre-trained. Section 6.4 contains a logical sequence of experiments showing that:

- Trainable filters can replace the encoder CNN in an already pre-trained model for fine-tuning.
- A physiologically adaptable front-end performs as well as a CNN in a pre-trained model.
- Trainable filters can be incorporated during self-supervision.
- When trained with a large transformer model, SincNet filters do not tend to learn wide-band filters as they do with a smaller MLP model.

6.2 Background

6.2.1 Self-Supervised Models

Foundations

Self-supervised learning arose around 2008 in the NLP field with the model of (Collobert & Weston, 2008). Self-supervision differs from supervised learning in the way the loss function is computed. Supervised learning necessarily requires labeled data; the output of the network is directly compared to the labels and the difference is back-propagated through gradients. Assigning labels can be onerous. By contrast, self-supervised learning has the advantage of needing no labels to train, meaning it can make use of huge amounts of data. The model is typically trained either as an auto-encoder (the data are the labels) or by contrastive loss (see below).

Self-supervised models rely on a two-stage training procedure: pre-training and fine-tuning. These map onto what might be traditionally called training and adaptation respectively. Pre-training is resource-intensive and the resulting models can be large. A significant recent trend has been for proponents to make such models available online (Ott et al., 2019). A core goal of this study is to investigate what can be done in the acoustic research field starting from those available pre-trained networks.

Wav2vec

Wav2vec (Schneider et al., 2019) was the first attempt at applying self-supervision in the sense of BERT to the ASR field. It was inspired by wav2letter (Collobert et al., 2016) and unsupervised machine translation (Lample et al., 2017), a model that translates between languages based on two unlabeled datasets. The structure of wav2vec (Schneider et al., 2019) comprises two main blocks: the encoder network and the context network. The encoder consists of a 5-layer CNN down-sampling the input waveform from 16 kHz to a 100 Hz signal. The context network in wav2vec is also a CNN with kernel sizes of three and strides of one. In vq-wav2vec, Baevski et al. (Baevski et al., 2019) proposed to integrate quantization between the encoder and the context layers. The recent Wav2vec2 (Baevski, Zhou, et al., 2020) parallelizes the quantization and an enhanced version of the wav2vec structure; the encoder CNN is now seven layers deep and down-samples the signal to 50 Hz and the CNN-based context layer is replaced by a transformer. Of those two networks, the transformer stack accounts for 94.4% of the total number of trainable parameters.

Wav2vec models use contrastive loss as an objective during the self-supervised training phase, a character-based label classification and connectionist temporal classification (CTC) of Graves et al. (Graves et al., 2006) during the finetuning phase, and a greedy CTC decoding algorythm for the test phase.

In the self-supervised training phase, the convolutional frontend and transformer-based encoder are trained through a contrastive loss mechanism. The original contrastive loss of wav2vec was formulated to favor predictability for adjacent observations, or “predictors”, with the opposite for more distant ones, hence “distractors”. Oord et al. (Oord et al., 2018) proposed another approach to the contrastive loss computation, this time based on the cross entropy loss. The loss equation in wav2vec2 adopts this more recent approach, being

$$\mathscr{L}_m = -\log \frac{\exp(sim(\mathbf{c}_t, \overbrace{\mathbf{q}_t}^{\text{predictor}})/\kappa)}{\sum_{\tilde{\mathbf{q}} \in \mathbf{Q}_t} \exp(sim(\mathbf{c}_t, \underbrace{\tilde{\mathbf{q}}}_{\text{distractor}})/\kappa)} \tag{6.1}$$

where *sim* is the cosine similarity, $sim(\mathbf{a},\mathbf{b}) = \mathbf{a}^T\mathbf{b}/|\mathbf{a}||\mathbf{b}|$, κ is a constant that regulates the entropy of the cosine similarity, preserving the relative ranks of events. In the machine learning field, the analogous parameter controlling the smoothness of probability distributions is commonly referred to as ’temperature’. The parameter c_t is the context representation and $q \in Q_t$ are vectorized samples of other parts of input waveform in the latent space. The goal of Equation (6.1) is to find a quantized representation for speech in the latent space, training towards orthogonal representations of the quantizers in that latent space. The model outputs a context representation c_t that is able to guess the true $\mathbf{q}_t$ vector quantization (the predictor) of the latent space out of $K+1$ candidates (with K distractor quantizers and one closely related target; in wav2vec2, 100 distractors are sampled out of the same utterance).

For finetuning, a projection layer is added on top of the transformer-based encoder. A character-based classification task then trains this projection layer to recombine the vectors learned in the latent space to predict characters then with CTC the characters are aligned with the transcripts to compute the loss from the difference between the probability prediction from the CTC encoding and the ground truth transcripts.

During the test phase, greedy CTC decoding is used to predict the transcripts of the evaluation speech data. The difference between CTC in the finetuning task and decoding phase is the output: during finetuning the output is a number which corresponds to the loss while during decoding the output is the decoded text.

6.2.2 Cochlear Models

Filterbanks

Cochlear models have been studied for many years, with our current understanding perhaps going back to the work of (Von Békésy, 1960). Simplistic (but functional) approaches consider the cochlea as a natural filterbank (Geisler, 1976; Zwislocki, 1953).

Comparatively recent studies have continued to seek biological plausibility. (Lyon, 2011a, 2011b, 2017b), for instance, proposed a model of the complete auditory path where the cochlea is modeled with a cascade of resonators. This model has since been implemented on an FPGA (Thakur et al., 2014) and used in applications such as speaker identification (Islam et al., 2022) and sound localization (Xu et al., 2021).

Of particular interest for speech processing is the filterbank scaling. Probably the best known frequency distribution is the mel scale (Pedersen, 1965), based on frequencies judged to be equally spaced in human perceptual tests. This has been ubiquitous in feature extraction algorithms for ASR, especially in its guise as MFCCs. The PLP of (Hermansky, 1990b) favored the Bark scale, based on noise bandwidths required to mask tones (Moore & Glasberg, 1983; J. O. Smith & Abel, 1999; Zwicker, 1961). Physical measurements are also possible; the greenwood (Sridhar D, 2006) and ERB (equivalent rectangular bandwidth) (Zwicker, 1961) scalings lead to more extreme warping than mel.

Current Understanding

Current explanations of the workings of the cochlea are as processes of wave propagation through an active oscillator system (De Boer, 1983). According to our current best understanding, the cochlea works as an array of Hopf oscillators (Hudspeth et al., 2010a; Hudspeth, 2008). These active oscillators incorporate the interaction between the inner and outer hair-cells with the tectorial and basilar membrane, and explain oto-acoustic emissions (Kemp, 2002; Probst et al., 1991) as arising from the outer hair cells. Several works have implemented those Hopf oscillators as models for the cochlea (Ammari & Davies, 2020; Hamilton et al., 2007, 2008).

Notwithstanding, in the present study we limit ourselves to filterbanks, with active systems being a goal for further research work.

6.2.3 ASR with Trainable Filters

From Cochlear Models to E2E ASR

For many years, ASR front-ends such as MFCC (Davis & Mermelstein, 1980) and PLP (Hermansky, 1990b) were loosely based on models of the cochlea. However, given the large number of choices within such models, and parallel success with raw images as input (Krizhevsky et al., 2012), E2E approaches have been investigated for audio processing. Early work involved trainable convolution layers on raw audio inputs (Hinton et al., 2012; Palaz et al., 2013b, 2015). Although they can perform well, such black box models lack interpretability, explainability, comprehensibility and transparency (Chakraborty et al., 2017; Rudin, 2019). Retaining an explicit filterbank structure can alleviate these issues. Candidates for the filterbank have included Gamma-tone (López-Espejo et al., 2021; T. Sainath et al., 2015), Gabor (Noé et al., 2020; Zeghidour et al., 2018, 2021), SincNet (Ravanelli & Bengio, 2018a, 2018b), and Spline filters (Balestriero et al., 2018). Zeghidour et al. (Zeghidour et al., 2018) in particular showed that using trainable filters consistently increased the performance of ASR compared to a model where MFCCs were used. Since we do not have a hypothesis that some trainable filter models would perform better, that this work builds on a previous work using SincNet filters in a previous study, and that there are computational advantages of doing so (see Section 6.2.2), we take the SincNet model of (Ravanelli & Bengio, 2018b) as representative of the above models for the present study.

SincNet

SincNet is characterized by a convolutional filter as the first layer in a larger convolutional front-end. In the initial SincNet implementation, this convolutional front-end was followed by a five-layer MLP, whereas in this work, the main idea is to combine this encoder with pre-trained transformers. The implementation of the filter layer uses the Fourier transform property: a convolution in the temporal domain corresponds to a multiplication in the frequency domain. Thus, using a sinc filter as kernel in a convolutional layer gives the filtered signal as output. The name arises because the Fourier transform of a rectangular filter in the frequency domain corresponds to $\text{sinc}(x) = \frac{\sin(x)}{x}$ in the temporal domain. SincNet is implemented as a combination of two low-pass filters $F_1 - F_2$ with cut-off frequencies $f_1 > f_2$, giving a band-pass filter of bandwidth $f_1 - f_2$ and a central frequency of $\frac{f_1+f_2}{2}$. The central frequency and bandwidth are the two trainable parameters. In the temporal domain, the filter is then given by:

$$h[n] = \text{sinc}(2\pi f_2 n) - \text{sinc}(2\pi f_1 n) \tag{6.2}$$

Ravanelli and Bengio (2018b) showed that SincNet outperformed MFCCs. They also showed that using SincNet filters gave better results than using only CNN layers. A second study by the same authors showed that the proposed architecture converges faster, performs better, and is more interpretable than standard CNNs (Ravanelli & Bengio, 2018a). They showed that such trainable filters could map onto specific speech-related features like formants, while standard CNNs did not. SincNet was subsequently incorporated into a joint CTC-attention training scheme by Parcollet et al. (2020). The authors showed that their approach outperforms previously investigated E2E systems.

Based on these examples in the literature that show the reliability of SincNet as a front-end for ASR, we use this trainable filterbank in this work.

6.2.4 Speech Features

We are particularly interested in the interpretability of the learned filterbanks. Pitch and formants are well known important speech features. For human speech, pitch typically lies between 85 Hz and 300 Hz. Formants are resonances of the vocal tract and define vowel sounds. The first formant is generally situated between 200 Hz and 800 Hz; the second formant lies between 500 Hz and 2500 Hz. (Olive et al., 1993)

6.3 Method

6.3.1 Overall Hypothesis

In Chapter 5, we showed that a model of the cochlea trained in an E2E manner behaves unexpectedly. The model learns a combination of expected narrow-band structures representative of the cochlea. However, it also learns wider-band structures that are more representative of the higher level auditory pathway. The network used was a combination of a SincNet front-end with a simple MLP as context network and trained in a supervised manner. The state-of-the-art in neural models for acoustic processing of speech is now associated with pre-trained stacks of transformers. Such stacks have been shown to exhibit behavior quite similar to that of the human brain using functional Magnetic Resonance Imaging (fMRI) (Millet et al., 2022).

In the following experiments, we aim to show that a plausible model of the cochlea can be combined with an otherwise black-box pre-trained model to yield a model much closer to the human auditory pathway. If such a model continues to behave in the same way as our simplistic one, we could further conclude that something is wrong with our cochlear model; certainly we have no biological evidence for wide-band structure at such a low level. By contrast, if the new model behaves differently, we could conclude that our simplistic model is too simplistic, and that a larger (implying pre-trained) model is *necessary* for such low level auditory investigation.

6.3.2 Pre-Trained Model

The model used in this work is based on wav2vec2 (Baevski, Zhou, et al., 2020), presented in Section 6.2.1. For this study, the encoder CNN of wav2vec2 is replaced with SincNet. The encoder layer is composed of SincNet filters followed by a classical CNN of 3 layers as in the SincNet baseline; this CNN down-samples the input signal to 50 Hz, which is then compatible with the further transformer layers. It is illustrated in Figure 6.2. The training follows the wav2vec2 method, but is customized to take the SincNet filters into account without starting the pre-training from scratch.

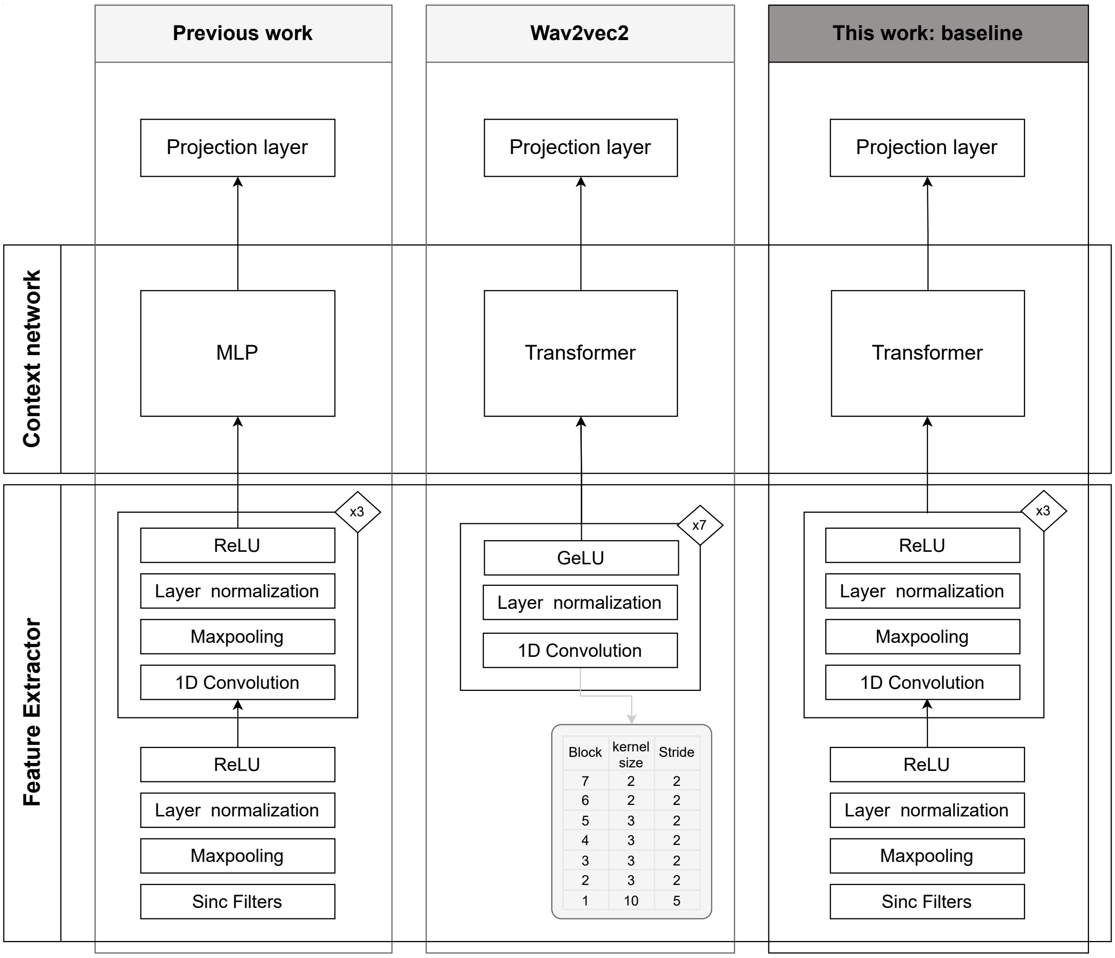
The framework used is FAIRSEQ (Ott et al., 2019), (which stands for Facebook AI Research in sequence modeling), a framework developed and used by Meta, which has an implementation of wav2vec that can be easily modified. The modifications are described in following sections. Moreover, the code is adapted to run on several Graphics Processing Unit (GPUs) in parallel.

6.3.3 Experimental setup and training protocol

We use the wav2vec2 (Baevski, Zhou, et al., 2020) network of the fairseq framework (Ott et al., 2019). To be able to adapt separately the different network parts with different learning rates we use the *composite* optimizer mechanism with an overall *pass_through* learning rate scheduler, since the different networks will have a specific learning rate scheduler assigned. For the convolutional downsampling part of the network as well as the projection layer, we use a *polynomial_decay* scheduler: with a short warmup stage and a long polynomial decay stage. The convolutional part of the network is initialized wit a learning rate of $6 \cdot 10^{-4}$. For the projection layer, the learning rate is initialized to $3 \cdot 10^{-6}$. For the transformer based part we created a *4-stage* scheduler: starting with a freezing stage where the learning rate is set to 0, then a warmup stage, a plateau and a linear decay stage. The optimizers are for all schedulers fixed to adam and the learning rate is initialized to $3 \cdot 10^{-6}$.

6.3.4 Dataset

We use the LibriSpeech dataset (Panayotov et al., 2015) for all experiments presented in section 2.3.4. We run each experiment either on the subset of 100h or the complete Librispeech dataset. For every experiment, the final result is computed on a single training run over several



Block	kernel size	Stride
7	2	2
6	2	2
5	3	2
4	3	2
3	3	2
2	3	2
1	10	5

Figure 6.2: A schematic overview of the original SincNet implementation model used as baseline in our previous work, the wav2vec2 fine-tuning path and the proposed fine-tuning path in this work. Based on compositionality capacity of networks, we combined the feature extractor of the original SincNet model with the pre-trained transformer of wav2vec2.

epochs. The variance is computed on the test using a Bayasian approach.

6.4 Experiments

6.4.1 Can Trainable Filters Replace the Encoder CNN in an Already Pre-Trained Model?

Our first hypothesis posits that the CNN encoder block acquires information learnable through trainable filters. To validate this, we replace the front-end with a SincNet encoder (see Figure 6.2) and compare post-training ASR performance to the baseline. Secondly, by incorporating modifications from our previous work, we expect improved results compared to the original SincNet implementation. Third, the trainable filters in the initial encoder layer should mirror patterns learned by the pre-trained encoder.

Choice of Global Parameters

In order to test this hypothesis, several implementation parameters are adapted: the learning rate scheduler, the structure of the new encoder, the kernel size of the filter layer and the number of updates of the model.

Learning Rate Schedulers

If a pre-trained model is used in, say, ASR, there is normally an adapter layer at the output of the pre-trained model. This layer can be trained whilst keeping the pre-trained model fixed by simply not back-propagating gradients further than the adapter. By contrast, the SincNet that we wish to train is an input of the pre-trained model; gradients must be back-propagated through the pre-trained model. The new weight value is given by the update parameter learning rule in Equation (6.3).

$$\theta_{t+1} = \theta_t + \Delta\theta_t \tag{6.3}$$

where θ_t is the weight value one time-step before the updated value. $\Delta\theta_t$ is dictated by the equations of optimizer update—in the case of this work, Adam (Kingma & Ba, 2014).

$$\Delta\theta_t \quad = \quad -\frac{\eta}{\sqrt{\hat{v}_t + \epsilon}}\hat{m}_t \tag{6.4}$$

where:

- $\hat{m}_t$ and $\hat{v}_t$ are bias-corrected estimates of the linear combination of the gradient with first and second moment estimates.
- ϵ is a small constant preventing a division by 0.
- η is the learning rate.

To freeze the model weights, we set the learning rate to zero. According to the gradient descent update Equation (6.4), if the learning rate η is set to zero, the parameters to which this 0 learning rate is applied are frozen to their initial value, which is a common machine learning practice.

In this work, we adopt a learning rate scheduler integrating a freezing period. During this freezing period, the encoder network is able to learn what the previous CNN layers had learned during the pre-training phase and the context network stays frozen.

The original wav2vec2 implementation uses two types of learning rate schedulers: the polynomial decay and tri-stage learning rate scheduler. The polynomial decay scheduler is used for non-pre-trained network chunks; in the wav2vec framework, this corresponds to the self-supervision phase while the tri-stage learning rate scheduler is used for fine-tuning. For this experiment, the polynomial decay learning rate was assigned to the encoder network (Figure 6.3c), since this part of the network is replaced by a SincNet encoder and consequently trained from scratch in the experiment. For the context network, we created a hybrid version of the tri-stage learning rate scheduler incorporating a freezing period at the beginning (Figure 6.3d). The whole training scheme is schematically summarized in Figure 6.4, clearly depicting the two training phases. Freezing the transformers and projection layer in the first place avoids catastrophic forgetting of the transformer weights. FAIRSEQ's composite optimizer enables distinct learning rate schedulers for different model parts, utilizing a pass-through general optimizer to automatically associate each part with its specific scheduler and optimizer.

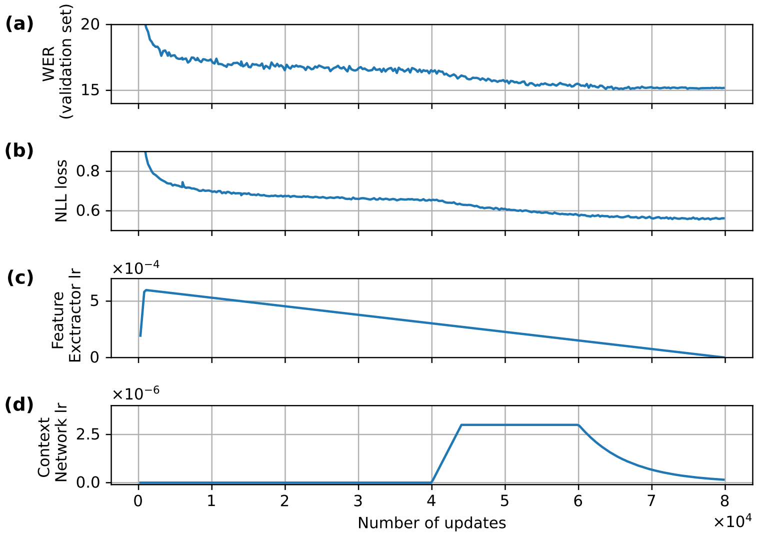


Figure 6.3: Training curve of the long-run training of 100 k updates. (**a**) depicts the WER of the validation set (dev-other), (**b**) contains the negative log likelihood of the loss, and (**c**,**d**) depict the learning rate evolution of the feature extractor and context network, respectively.

SincNet Modifications

In the previous work (section 2.5), we showed that removing the maxpooling layer after the filters was physiologically relevant and improved the performance. In this experiment, we compare the performance of the initial SincNet structure and the adapted structure.

Kernel Size

With the larger model, we observed a saturation effect at lower frequencies in a draft experiment, linked to the maximum filter precision dictated by the kernel size. We replaced the initial kernel size of 129 by 400 in the sinc filter initialization; this both corrected the above issue, and

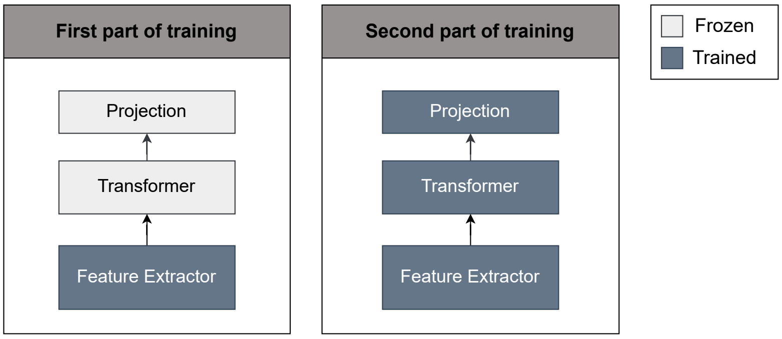


Figure 6.4: A more schematical explanation of the behavior of the network training explained in Figure 6.3. There are 2 training phases: one up to 40 k updates where the whole model except the feature extractor is frozen, and the second phase, where everything fine-tunes together.

corresponded to the 25 ms used as window size in classical MFCC computations.

Number of Updates

An update corresponds to a back-propagation of gradients in the model, which updates the parameters, while an epoch corresponds to passing the whole dataset through the model. For most of the experiments, the number of updates is fixed to 10 k, which corresponds to 38 epochs with the training set train-clean-100 for the fine-tuning part. For completeness and comparison, we also report one experiment where the model was able to train for 100 k updates, with 40 filters and a kernel size of 400.

Results

The results of this first experiment are summarized in Table 6.1 and Figure 6.3. Concerning the performance, the WER results of Table 6.1 are all below 4%. This means that a SincNet encoder is capable of replacing the original wav2vec2 encoder when keeping the context network fixed.

Table 6.1: Comparison between a short and long run using no maxpooling after the filter layer in the first 2 experiments and with a maxpooling that has a kernelsize of 3 in the third experiment. Thus, the third experiment has a downsampling of a factor 3 just after the sinc filters via a maxpooling operation in the time domain.

n_filters	Maxpooling	Kernel Size	n_updates	WER
40	-	400	100 k	**3.31**
40	-	400	10 k	3.53
40	3	400	10 k	3.64

The encoding and impact of the learning rate scheduler on the WER and loss curve are illustrated in Figure 6.3. During the first half of the training updates, the context network is frozen through the zeroing of learning rate, while the encoder network is able to train. In the second half of the experiment, the training of the context network is enabled; the transformer can slightly adapt itself to the trained encoder. In the WER and loss curves, we can see a clear impact of the transformers when they obtain the ability to train around 40 k updates. However, when only enabling the SincNet encoder structure to train, the network already shows a decent performance during the first part of the training.

As summarized in Table 6.1, three different experiments were performed to address this first hypothesis: two short experiments to analyze the impact of removing the maxpooling layer and a third experiment where the training time was much longer. As in the previous work (section 2.5), removing the maxpooling function improves the global performance of the ASR. Further, letting the experiment train for a longer period makes the performance increase.

The shape of the filter distribution for the different runs of Table 6.1 are illustrated in Figure 6.5. When training the network for 100 k updates instead of 10 k, the WER continues decreasing, but the spatial distribution of the filters does not move much anymore. Changing the network structure by considering the maxpooling layer, however, has a small impact: for lower frequencies, some wider filters are appearing. This means that using maxpooling precludes the wide-band filters to be learned higher up in the network; nevertheless, the proportion of wide-band filters are much lower than what we obtained in the previous work, and the size of the bandwidth is smaller.

A consistent pattern emerges in the filter distribution shape across various runs within the spectral content, up to approximately 4 kHz. Below 1 kHz, a bump of very small (and consequently precise) filters occurs in the filterbank between 200 Hz and 800 Hz; this frequency range corresponds to the first formant (see Section 6.2.4). From 1 kHz to 5 kHz, the bandwidths of the filters are all around 400 Hz. Above 5 kHz, the filters seem to learn something different in every run.

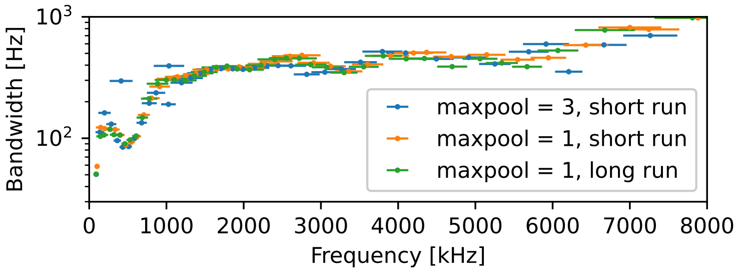


Figure 6.5: Comparison of filter distribution for different run lengths and internal SincNet structure. Using maxpooling within the filter structure makes some wide-band filters appear in low frequency ranges.

6.4.2 Does a Physiologically Adapted Front-End Perform as Well as a CNN in a Pre-Trained Model?

Hypothesis

(Ravanelli & Bengio, 2018a) showed that using the SincNet model instead of simple CNN converged faster, performed better, and was more interpretable. We hypothesize that if we train a SincNet model and a CNN model with the same number of layers from scratch, combined with a pre-trained context network, the SincNet model would also show those characteristics.

This can be tested by performing and comparing four experiments: The first experiment retrains the wav2vec2 encoder network from scratch to gauge its performance without the pre-training information, setting a baseline for encoder relearning while keeping the pre-training information of the context network. Second, we use the baseline CNN with a layer of the same size as the trainable filters to be comparable with (Ravanelli & Bengio, 2018a), but this time combined with a pre-trained context network. The third experiment consists of training a large SincNet encoder. The baseline CNN structure is combined with a layer of trainable filters and trained with the pre-trained transformers. Finally, we compare these experiments with a SincNet structure containing fewer CNN layers (than the structure used in the experiments of Section 6.4.1).

Results

The **convergence speed** can be analyzed in the training curve. Figure 6.6 shows that in terms of training, the SincNet structure converges faster towards an equilibrium compared to a pure CNN structure.Concerning the validation, this is true in the very beginning, especially concerning the CNN with the same shape as SincNet (red curve) up to 2000 updates. We based our analysis of the convergence speed on the update metric to align the loss on the amount of data that go through a forward–backward pass. However, to be complete, Table 6.2 details the time that every experiment takes and the number of parameters that are contained in the front-end for purposes of comparison. **The performance** of the different experiments is summarized in Table 6.3. Overall, the 95% confidence interval of the large SincNet structure performance overlaps widely with the baseline experiment; this means that the performance with SincNet is not significantly better than the baseline CNN. However, between the SincNet

Table 6.2: Table summarizing the time of each experiment on 4 parallel GPUs and the number of parameters contained in the feature extractor. Note that this number of parameters is quite small compared to the rest of the network (90 M parameters).

Model	Time	N. Parameters
Small CNN	7 h 14 min	0.575 M
Large CNN	9 h 48 min	4.406 M
Relearn CNN (w2v2 shape)	11 h 57 min	4.206 M
Relearn CNN (SN shape)	14 h 7 min	4.422 M

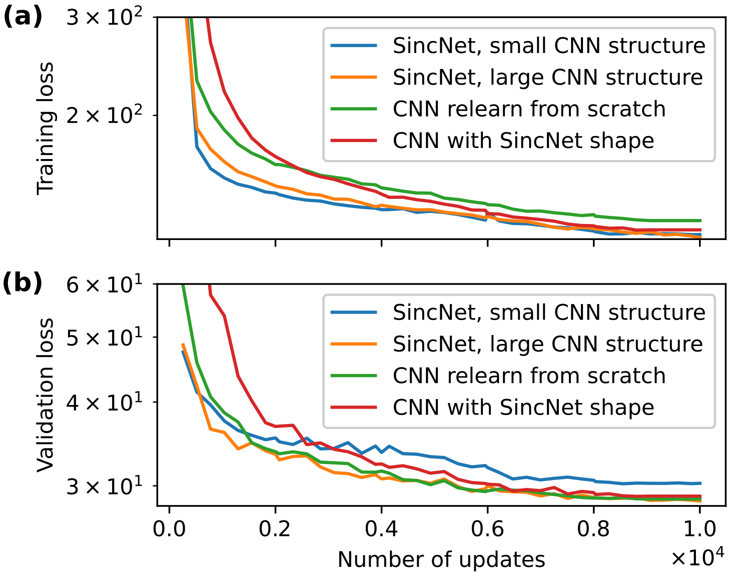


Figure 6.6: Loss curve during **(a)** training and **(b)** validation.

shape CNN and the SincNet implementation, which correspond to the experiments performed in (Ravanelli & Bengio, 2018a), the overlap is less important and SincNet performs slightly better.

Table 6.3: Performance of the different experiments on the capacity of SincNet (SN) to replace the initial CNN structure with a 95% confidence interval, assuming the result is beta distributed.

	Train Loss	Valid Loss	WER [%]
Relearn CNN (w2v shape)	129.5	28.67	3.35 ± 0.15
Relearn CNN (SN shape)	124.7	28.96	3.45 ± 0.15
SincNet (large CNN)	**120.8**	**28.48**	**3.33 ± 0.15**
SincNet (small CNN)	122.3	30.26	3.53 ± 0.15

Finally, concerning the **interpretability**, Figure 6.7 illustrates the normalized sum of the filters in the CNN implementation and SincNet after training.
In both cases, "bumps" occur around 700 Hz and 1.4 kHz, which correspond to typical first- and second-formant frequencies. In (Ravanelli & Bengio, 2018a), those interpretable elements in the normalized filter sum were more distinguishable in the SincNet structure than in the CNN structure. Further, for frequencies above 4 kHz, the SincNet filters more efficiently reduce the amount of information recorded in that area than the CNN structure.

To summarize, in the context of a self-supervised model with a pre-trained context network, using a SincNet encoder converges faster and performs as well as the baseline CNN trained from scratch and better than the similar shape CNN. Concerning the interpretability, despite

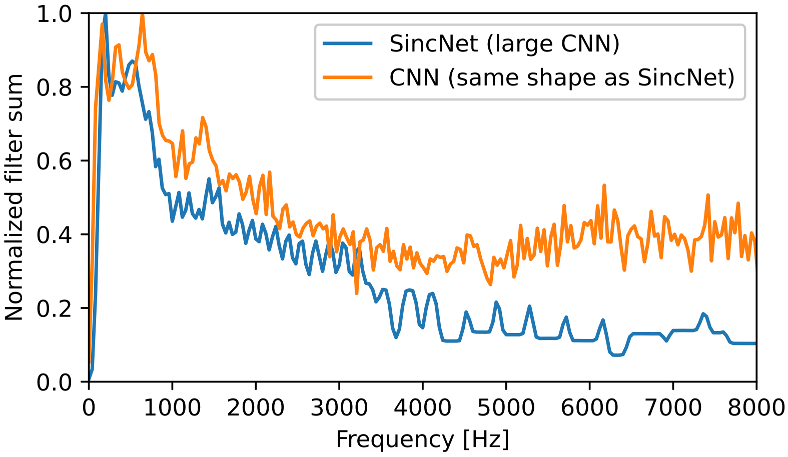


Figure 6.7: Comparison of the content of the normalized filter sum using SincNet filters and using a same-size generic CNN.

being more interpretable by design, the SincNet structure does not seem to better emphasize interpretable signal components than a CNN using 40 filters.

6.4.3 Can Trainable Filters be Incorporated during Self-Supervision?

Another way to train the filters using wav2vec2 is to incorporate the filters into the model in the self-supervision phase.

Hypothesis

Although more computationally demanding, using the self-supervision training path is more faithful to the original wav2vec2 training path and it enables the filters to train on two successive tasks. We hypothesize that continuing the self-supervision task while keeping the context network frozen would lead the filters to learn a similar distribution to the first experiment results. Moreover during self-supervised learning we can either let the filters train or keep them frozen and let them only train during fine-tuning. In this case, we expect the rest of the network to adapt itself to the frozen filter distribution and we do not expect the filters to differ much from what they previously learned during further fine-tuning. However, we expect the free filters to show slightly better performance since they are able to adapt to the fine structure in the frequency range.

Self-Supervised Learning

The first step of this experiment consists of further training the model through the contrastive loss task (see Equation (6.1)). To be consistant with the original self-supervision dataset we included the two other training subsets of LibriSpeech (see Table 2.2).

An important parameter to fix is the number of updates needed to obtain a comparable performance as the baseline pre-trained model. Since self-supervision implies a performance measure that is not based on labels, the best comparison measures are the loss and the accuracy.

Table 6.4 compares the accuracy and loss of the training and validation curves of the baseline and the wav2vec2 model integrating the trainable filters. Using 10 k updates, the accuracy and loss differ from 10–20% from the baseline loss and accuracy. With 100 k updates, the final loss and accuracy match the baseline results up to 1% of error.

Table 6.4: Comparison of the loss and accuracy of the baseline pre-trained model and the model after a further pre-training for both frozen and trainable filters. After 10 k updates, the model already performs well but not as good as the baseline, while after 100 k, the model reaches the performance of the baseline in terms of loss and accuracy.

	Number of Updates	**Accuracy [%]**		**Loss**	
		Train	**Valid**	**Train**	**Valid**
Baseline	0	61.1	64.6	2.10	1.96
Frozen filters	10 k	52.2	62.2	2.49	2.13
Trained filters	10 k	56.9	63.0	2.34	2.06
Frozen filters	100 k	60.5	66.5	2.13	1.87
Trained filters	100 k	**61.4**	**67.2**	**2.09**	**1.84**

Fine-Tuning

The second step consists of fine-tuning this self-supervised model. In this fine-tuning experiment, the learning rates do not have to be disentangled as we did in the first type of training path, since the whole model has been pre-trained.

Results

The final performance is given in Table 6.5. For a fine-tuning of 10 k updates, the performance is best when the filters have been through a self-supervision task before fine-tuning (see Table 6.1).

Table 6.5: Performance of model using trainable filters in pre-training (100 k updates) and fine-tuning (10 k updates).

Pre-Training Phase	**Fine-Tuning Phase**	**WER [%]**
Frozen filters	Trainable filters	3.40
Trainable filters	Trainable filters	**3.37**

Since during self-supervised training, the filters are able to train, they can be visualized both after pre-training and fine-tuning (Figure 6.8). For the trainable filters, below 1 kHz, a couple of filters become very narrow-band, then around 1 kHz, 2 kHz and 3 kHz, the filters show a narrow-band bump in their structure as if they sought slightly more fine-grained structures at those specific frequencies. Filters do barely move between during self-supervised training and fine-tuning. This means that the rest of the model is probably more inclined to move towards a better suited equilibrium to minimize the loss. Moreover, globally the shape of the filter distribution shows similar behavior between the two different training paths.

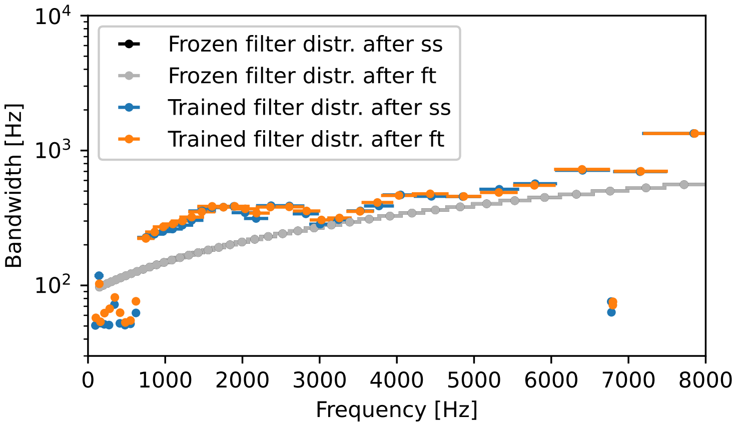


Figure 6.8: Filter distribution after the self-supervised (ss) training and fine-tuning (ft) of the wav2vec2 model. Globally, filters do not tend to move significantly during fine-tuning when they have been incorporated during pre-training.

In summary, using only supervised fine-tuning gives a broader flexibility for experiments where several parameters have to be changed, since only a fine-tuning run has to be adapted. Training through both self-supervision pre-training and supervised fine-tuning is more time and resource consuming, but better corresponds to the general idea of using both self-supervision and fine-tuning for training a model. Overall, the filters, when able to train before the transformer adaptation, tend to learn similar patterns.

6.4.4 Do Wide-Band Filters Appear in Some Other Training or Model Configurations?

An observation in conflict with our previous work (section 2.5) is that no wide-band filters appear within the current implementation. Figure 5.2 showed that some filters learned a very broad-band structure, while in Figure 6.5, for example, no such wide-band structures appear. Two hypothesis were apparent to explain why those filters could potentially not appear: the number of filters could be too low within the context of a self-supervised model and the freezing of the transformers by decoupling the learning rate scheduler could possibly preclude those filters from appearing.

The Hypothesis of Too Few Filters

Hypothesis

The first hypothesis suggests that a limited number of filters may hinder the emergence of wide-band structures. Additionally, the absence of maxpooling precludes the appearance of wide-band filters. This hypothesis can be tested by increasing the number of filters. If the number of filters is insufficient for the manifestation of wide-band filters, a larger number of them should lead to the emergence of these filters.

Results

Figure 6.9 illustrates the structure that the filters learn when initialized to 100, 80, 60, and 40 filters. Globally, below 1 kHz, filters learn a very narrow-band frequency-specific filters, above 1 kHz, filters have a bandwidth around 400–500 Hz for all the different numbers of initialized filters. Compared to the previous work, below 1 kHz, the number of very narrow-band filters is much higher in this experiment.

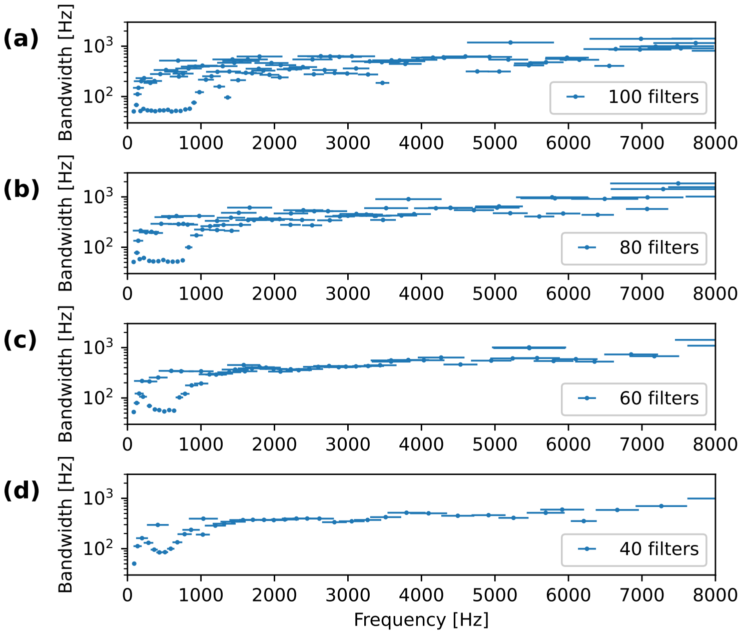


Figure 6.9: Distribution of the different filters in the frequency domain. The dots represent the central frequencies and the lines represent the bandwidths, similarly to Figure 5.2. The number of initial filters are (**a**) 100, (**b**) 80, (**c**) 60, and (**d**) 40.

A few wide-band filters do appear when the model is initialized with a high number of filters (80–100), For a smaller number of filters, (40, for example), the wide-band structures as we had in our previous work (section 2.5) do not appear anymore. Concerning the convenient number of filters to use based on this filter distribution analysis, 40 filters seem to be a convenient number to describe the whole frequency range. This corresponds with the conclusions of our previous paper and it is consistent with choices made in the literature, e.g., (Zeghidour et al., 2018).

To be complete, we also computed the performance for this model with the different numbers of filters, summarized in Table 6.6. Similarly to the observations made in our previous work (Coppieters de Gibson & Garner, 2022), the performance increases slightly with the number of filters we initialize.

Table 6.6: Performance on the dev-clean subset of LibriSpeech for different numbers of filters.

n_filters	Maxpool	Kernel Size	WER
40	3	400	3.64
60	3	400	3.61
80	3	400	3.57
100	3	400	3.56

The Transformers Precluding the Filters to Learn Wide-Band

Hypothesis

The second hypothesis for why the wide-band filters do not appear is linked to the training path: the transformers are first frozen before being able to train in parallel to the filters, while in the previous work, the whole network trained together. Until now, all experiments have been conducted by first keeping the transformers frozen for a given amount of time in order to train only the encoder network while keeping the transformers fixed. In order to verify this hypothesis, we perform an experiment where all learning rate schedulers start the warm-up period at the same time.

Results

When filters and transformers are free to train jointly from the pre-trained transformer version, the filter distribution does not learn wide-band structures and, moreover, it shows a similar distribution to previous experiments (Figure 6.10). This means that, used with a pre-trained transformer, the filters do not tend to get a wide-band structure as we had in (section 2.5).

In summary, when incorporating SincNet into a self-supervised model, the obtained filter distribution does not correspond to the results of our previous work. Only a very small number of wide-band filters appear when enlarging the total number of filters. Besides, the experiments confirmed that the number of filters needed to cover the frequency spectrum in ASR tends to 40. We conclude that a pre-trained context network probably encodes the combination of those wide-band structures, precluding those structures from appearing on the trainable filter layer.

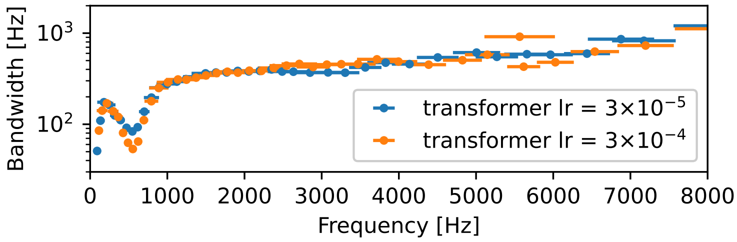


Figure 6.10: Filter distribution after training for 40 filters.

6.5 Conclusions

This study proposes an integration of two physiologically grounded concepts: trainable filters and self-supervision. It begins by delineating these concepts in Section 6.2 and subsequently elaborates on their joint training using a self-supervised context network while concurrently training a new front-end in Section 6.3. Section 6.4 outlines a logical sequence of experiments aimed at exploring the behavior of the trainable filters within this framework.

In the first experiment, we demonstrate that a new encoder can be trained while retaining the information acquired during pre-training in the transformer, avoiding a catastrophic forgetting of the transformer weights and resulting in performance close to the state of the art. Additionally, substituting the CNN encoder with a SincNet encoder sheds light on the information expectation of the transformer from the CNN, emphasizing the interest in frequency information at the trainable filter layer.

The second experiment illustrates that, when trained from scratch, the SincNet encoder converges more rapidly than the CNN. However, there is no significant performance improvement, and interpretability across the entire frequency spectrum does not reveal more speech artifacts compared to a CNN.

In the third experiment, we observe that the sole fine-tuning training path offers greater flexibility and is better suited for conducting experiments. While the pre-train–fine-tune training path aligns more with the original concept of self-supervision, it necessitates a larger dataset and substantially more time for the pre-training phase.

Lastly, in the fourth experiment, we investigate why wide-band filters cease to emerge. Through several experiments analyzing behavior with additional filters and employing different training paths, it appears that the pre-trained context network inhibits the emergence of wide-band filters in the initial layer.

Overall, this study demonstrates the feasibility of integrating and fine-tuning state-of-the-art networks with physiologically plausible models. Furthermore, the utilization of decoupled learning rate schedulers enables the fine-tuning of specific parts of the model. We posit that implementing more intricate physiological models and leveraging this approach can facilitate a deeper understanding of how physiological mechanisms may evolve to better interpret external inputs in the encoder network.

7 Integration of Hopf oscillator module into an ASR system

As illustrated on figure 7.1, this chapter brings together the knowledge of cochlear models and ASR technology. Building on the conclusions presented in Chapter 4, we describe the implementation of the Hopf oscillator into a module, simplifying the differential equations system from Section 4.3.2 to make it computationally compatible with ASR training.

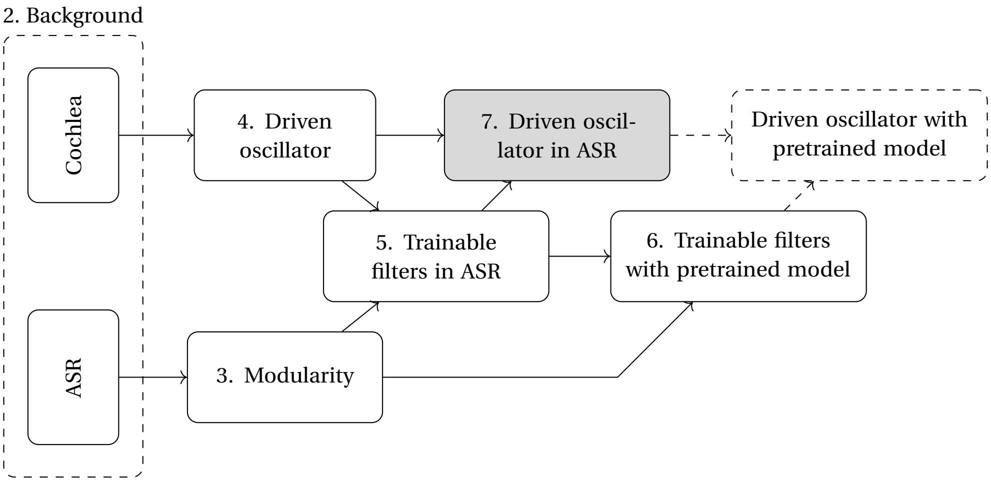

2. Background
Cochlea
ASR
4. Driven oscillator
7. Driven oscillator in ASR
Driven oscillator with pretrained model
5. Trainable filters in ASR
6. Trainable filters with pretrained model
3. Modularity

Figure 7.1: General overview of the thesis.

The primary goal of this chapter is to train an ASR system using a Hopf oscillator module as its frontend. This is achieved through two main steps: first, a simple integration of the Hopf module; and second, an integration of the efferent path through a larger feedback implementation in the encoder module.

Furthermore, we propose a more comprehensive analysis of performance within a noisy context. The adaptation mechanism of the auditory path exhibits interesting noise robustness capacities.

7.1 Background

The background material for this chapter has been covered in previous chapters.

- Hopf oscillators are known to model the active amplification mechanism of the cochlea. A broad literature review has been presented in chapter 4.
- ASR systems have evolved from statistical based HMM models to self-supervised models. This has become possible through the advent of increasing computational capacities of modern hardware and innovations in neural networks. For computational reasons, we use the ASR framework of chapter 5.

7.2 Learning parameters

In deep learning, parameters are optimized using the gradient descent algorithm, which iteratively adjusts model weights to minimize error. This method shares similarities with neural adaptation: synaptic strengths and types are shaped by neurotransmitter flow, which can be modelled by the gradient descent mechanism.

However, some parameters need a real-time adaptation even during the testing phase, which cannot be learned by traditional gradient descent algorithms. Instead, these adaptations are governed by a gradient computation based on differential equations that dynamically adjust to the evolution of incoming signals, whereas classical systems compute the gradient through the loss propagation. The cochlear model, which requires rapid parameter updates to accommodate dynamic sound waveforms, exemplifies the need for adaptive models that can learn from experience

In this work we employed the differential equation training for the Hopf implementation module, mimicking the behaviour of the cochlea with its active gain control loop. For the rest of the auditory path and the efferent path feedback, we use gradient descent as learning mechanism. This task division of the learning adaptation is illustrated in figure 7.2.

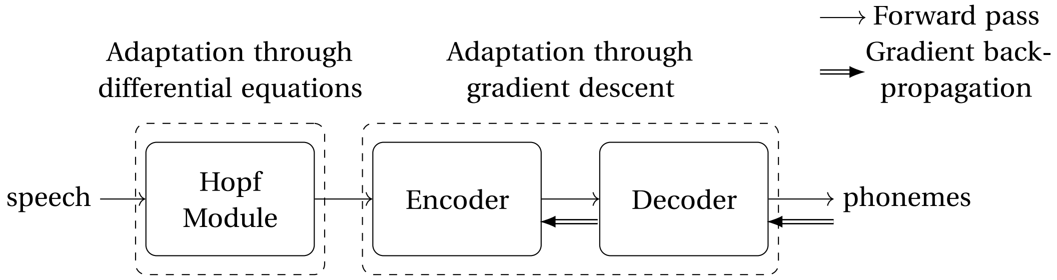

Forward pass
Gradient back-propagation
Adaptation through differential equations
Adaptation through gradient descent
speech
Hopf Module
Encoder
Decoder
phonemes

Figure 7.2: Learning mechanism: division of different learning routines over the different parts of the model according to the physiological equivalent.

7.3 Hopf oscillator module

7.3.1 Mathematical formulation

Our model mainly focuses on the general oscillation of the cochlea with a special focus on its active amplification characteristics and tuning at the Hopf bifurcation. Therefore, we mainly build our mathematical model based on literature that utilizes the Hopf equation (which are presented in section 4.1.4). As a reminder, we use the Hopf equation 4.5 as a baseline and enhance by adapting the frequency scaling and adding a tuning equation of the bifurcation parameter μ.

Adapted frequency scaling

The proposed system of equations with a frequency scaling parameter f is given by:

$$\begin{cases} \frac{1}{f}\dot{r} = \mu r + \beta r^3 + F\cos\theta \\ \frac{1}{f}\dot{\theta} = \frac{2\pi f_c}{f} - \frac{F}{r}\sin\theta \end{cases} \tag{7.1}$$

The frequency scaling equation proposed by Stoop et al. (2016) relies on a pure logarithmic scaling, while we are interested in other types of scalings (mel, greenwood, etc.). This reveals that the f-variable is directly linked to the bandwidth that the oscillator filters will have with a relationship given by: $f \sim f_{bw}$. The global equation thus becomes:

$$\begin{cases} \frac{1}{af_{bw}}\dot{r} = \mu r + \beta r^3 + F\cos\theta \\ \frac{1}{af_{bw}}\dot{\theta} = \frac{2\pi f_c}{af_{bw}} - \frac{F}{r}\sin\theta \end{cases} \tag{7.2}$$

Moreover, the separation of f_c and f_{bw} in the Hopf equation enables a better control over the filter distribution.

The different parameters are defined as:

- β : The first Lyapunov coefficient that needs to be negative in order to have a stable solution over the whole domain.
- f_c and f_{bw}: Respectively the central frequencies and bandwidth of the different oscillators defined based on mel-scale.
- a : a scaling factor between f and f_{bw}
- r, θ and μ: the parameters defining the system respectively the radius, anlge and bifurcation parameter of the different oscillators.
- F : The input signal corresponding to speech for our experiments

Tuning the bifurcation parameter μ

The tuning of the bifurcation parameter can be implemented in two different ways. One approach is to use a differential equation with a feedback mechanism, based on the implementation proposed by Camalet et al. (2000):

$$\dot{\mu} = \frac{\mu_{max} - \mu}{\tau}\left(1 - \frac{F^2}{\delta^2}\right) \tag{7.3}$$

Alternatively, an equivalent feedforward mechanism based on the value of the input signal F also provide a good estimate of the best suited new value for μ.

$$\mu_{th} = \mu_{\max}\left(1 - \frac{x^2}{\delta^2}\right) \tag{7.4}$$

Where the different parameters are defined as:

- $\mu_{\max}$: The maximum value fixed for the bifurcation parameter.
- δ: The bifurcation threshold, defining the limit between the damping and active amplification mode.

A linear combination of the computed theoretical value and the previous value of μ gives a smooth transition to the new equilibrium that corresponds to the damping or amplification value dictated by μ_{th}. Another aspect of the ear is the limitation of the damping or active amplification. Therefore, a tanh is wrapped around this function to limit the mu value between [-1, 1]. Introducing a smooth variation by keeping track of the previous value of μ leads to following equation:

$$\mu(t) = \mu(t-1)p + \tanh(\mu_{th}(t))(1-p) \tag{7.5}$$

With p being a fixed value that determine the proportion of the previous value of μ in the computation of the new value of μ combined with the theoretical computed value.

7.3.2 Implementation of the Hopf module

Figure 7.3 represents the module implementation of the system of equations. The model is composed of two main blocks: one computation block that computes the first derivative of the different parameters and applies the update step and one memory block keeps track of the last computed values of the variables which then generates the oscillator output. The oscillator is triggered by an incoming waveform and the output is computed based on the state of these parameters. The values of r, θ and μ are updated at every time step Δt.

First the first order derivatives of the different parameters with respect to those parameters is computed:

$$\begin{cases} \dot{r} = (\mu r + \beta r^3 + F\cos\theta)\pi f_{bw} \\ \dot{\theta} = 2\pi f_c - \frac{F}{r}\pi f_{bw}\sin\theta \\ \end{cases} \tag{7.6}$$

One potential issue with the system is that it may diverge to infinity if r tends to 0. This problem was mitigated by introducing a small ϵ with the same sign as r.

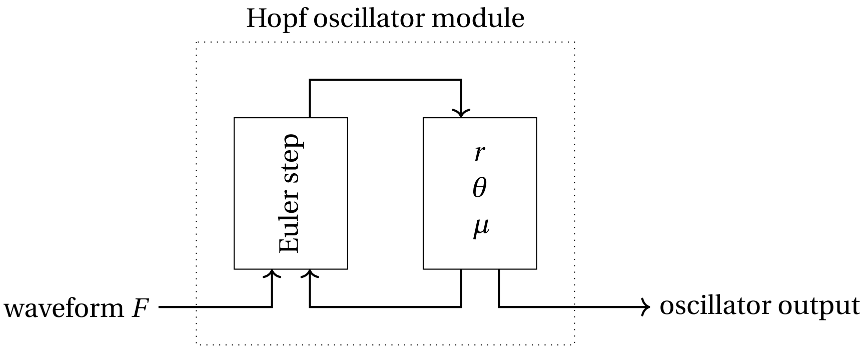


Figure 7.3: Hopf oscillator module schematic the oscillator output corresponds to the real value of the oscillator $r\cos\theta$.

A type of module compatible with this implementation is the recurrent neural network (RNN). Differential equations can be integrated into recurrent neural networks by computing every next step with the first-order method of Euler:

$$x[n+1] = x[n] + \dot{x}[n] \cdot \Delta t \tag{7.7}$$

Applied to the different parameters results in following system of equations:

$$\begin{cases} r[t+\Delta t, t+2\Delta t] = r[t, t+\Delta t] + \dot{r} \cdot \Delta t \\ \theta[t+\Delta t, t+2\Delta t] = (\theta[t, t+\Delta t] + \dot{\theta} \cdot \Delta t)\%2\pi \\ \mu[t+\Delta t, t+2\Delta t] = p\mu[t, t+\Delta t] + (1-p)\tanh(\mu_{th}[t+\Delta t, t+2\Delta t]) \end{cases} \tag{7.8}$$

7.3.3 Characterization of the Hopf Module

The Hopf model provides a mathematical description of the inner workings of the organ of Corti. It incorporates both a cube root compression and the active amplification mechanism that characterizes the interactions between the OHC and IHC by adapting the μ parameter to the incoming signal.

These characteristics represent a key difference from classical filterbanks. The combination of the cube root compression with the Hopf adaptation mechanism is crucial for explaining why the human ear has such a large amplitude range, allowing it to perceive a wide range of sounds. The amplitude of the output of the oscillator is directly related to the value of the parameter r in the differential equation system, which corresponds to the amplitude of the OHC.

This parameter r is directly influenced by the gradient $\dot{r}$ calculated at every step.

$$\dot{r} = (\mu r + \beta r^3 + F \cos\theta)\pi f_{bw} \tag{7.9}$$

To characterize the model, we analyse the values to which the amplitude r converges in the following scenarios:

- When there is no external signal ($F = 0$), which illustrates the active amplification mechanism.
- When there is no adaptation ($\mu = 0$), which isolates the cube root compression mechanism.
- Without any constraints on F and μ: under these conditions, the μ parameter tracks the system's amplification, diverging from a simple cubic root relationship.

We then compare the output of Hopf oscillators to that of a rectangular filterbank based on the SincNet implementation.

Active amplification mechanism with no incoming signal

When no signal is coming into the oscillator system, equation 7.9 becomes:

$$\dot{r} = (\mu r + \beta r^3)\pi f_{bw} \tag{7.10}$$

At equilibrium ($\dot{r} = 0$) the oscillators should converge towards an amplitude of $r = \sqrt{-\frac{\mu}{\beta}}$, except if $r = 0$ and no disturbance noise is added, then the signal stays at the unstable solution point $r = 0$. For a system with four Hopf oscillators (with the central frequencies: 365 Hz, 1260 Hz, 2907 Hz and 5937 Hz for oscillators 1, 2, 3 and 4 respectively) and $\mu = 1$, $\beta = -100$, the signal tends to the theoretical value $r = 0.1$ as shown in Figure 7.4.

When the bifurcation parameter μ is initialized to 0 with no input signal and able to adapt, it will converge to μ_{max}. The convergence value of the radius of the oscillators is then defined by $r = \sqrt{-\frac{\mu_{\text{max}}}{\beta}}$ and the radius takes a longer time to converge to that value as illustrated in Figure 7.5. Besides, we notice that oscillators with a smaller bandwidth take a longer time to learn this new amplitude than large-bandwidth filters.

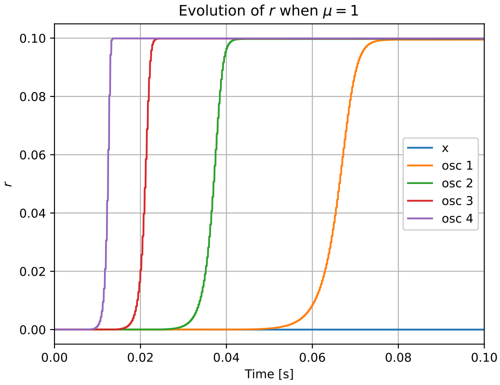


Figure 7.4: Evolution of the r-value in the oscillator when $\mu = 1$ is fixed

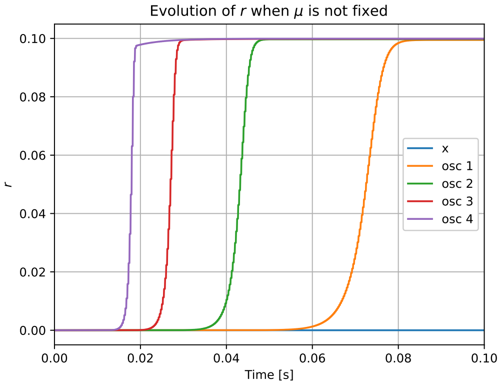


Figure 7.5: Evolution of the r-value in the oscillator when μ is initialized to 0 and evolves over time for μ_{max} fixed to 0.

Cube root compression

When the bifurcation parameter is fixed at the bifurcation point, a cube root relationship between the amplitude of the input signal F and the oscillator response r appears at equilibrium.

In this configuration, the differential equation of the oscillator amplitude $(\dot{r})$ equation becomes:

$$0 = \beta r^3 + F \cos\theta \tag{7.11}$$

The value that r tends to adjust to the value of the incoming signal $F = A\cos\theta'$. When the two signals are in phase, $\theta' = \theta$

$$r^3 = -\frac{1}{\beta} F \cos\theta \tag{7.12}$$
$$r^3 = -\frac{1}{\beta} A \cos\theta' \cos\theta \tag{7.13}$$
$$r^3 = -\frac{1}{\beta} A \cos^2\theta \tag{7.14}$$

This value is not constant due to the cosine, to find the theoretical value this equation tends to, we need to take the mean value of $Acos^2(\theta)$.

$$r^3 = -\frac{1}{2\pi} \int_0^{2\pi} \frac{A}{\beta} \cos^2\theta d\theta \tag{7.15}$$
$$r^3 = -\frac{A}{2\beta} \frac{1}{2\pi} \int_0^{2\pi} 1 + \cos(2\theta_i) d\theta \tag{7.16}$$
$$r = \sqrt[3]{-\frac{A}{2\beta}} \tag{7.17}$$

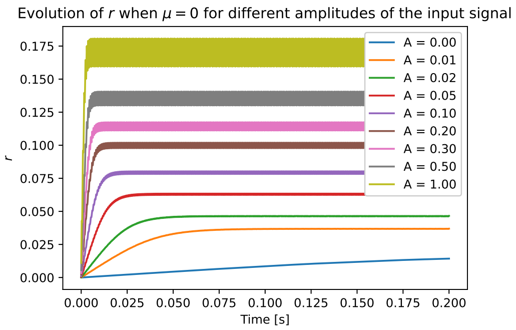


Figure 7.6: The evolution of the radius r of an oscillator for different signals tuned at its central frequency.

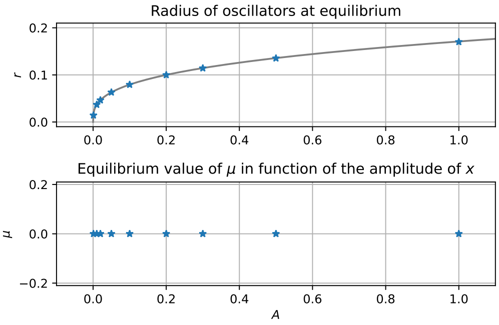


Figure 7.7: Cube root relationship between the input signal amplitude (A) and the oscillator radius (r).

Figure 7.6 shows the amplitude evolution of an oscillator for inputs signals with different amplitudes (A) and tuned at the resonance frequency of the oscillators. Figure 7.7 shows the final mean amplitudes transposed to the r-A plot. This shows that the oscillator output signals conform to the theoretical cube root relation. In addition, oscillators adapt more rapidly to signals with higher amplitudes.

For frequencies outside the bandwidth range, the oscillator fails to synchronize with the input signal frequency, exhibiting passive behaviour instead, which results in oscillations at lower levels for higher frequencies as illustrated in Figure 7.8. The response of the oscillator exhibits

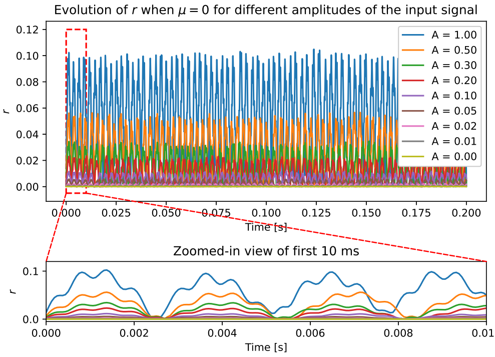


Figure 7.8: Evolution of the radius of an oscillator radius for in input signal tuned at a frequency outside of the frequency bandwidth of the oscillator with a zoom showing the transient.

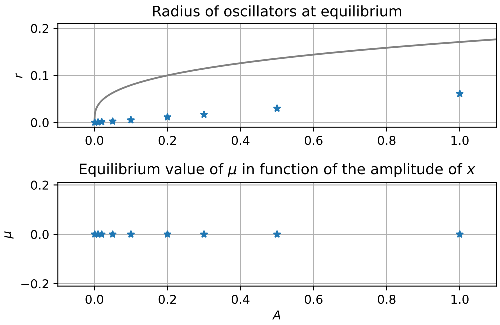


Figure 7.9: Linear relationship between the input signal amplitude (A) and the oscillator radius (r).

a linear relationship with the input amplitude (see Figure 7.9). The farther the frequency is from the oscillator's bandwidth, the less it will influence the oscillator's radius. This also reflects the coupling behaviour between neighbouring oscillators on the basilar membrane.

Adaptation of the bifurcation parameter μ to signal amplitude A

When there is no constraint on the input signal and the bifurcation parameter, the relation between the signal amplitude and the oscillator radius slightly diverges from the cube root compression due to the adaptation of the μ parameter. This is illustrated in Figures 7.10 and 7.11. For low amplitudes the converging value of r exceeds the theoretical curve, whereas for high amplitudes the signal is further damped down. The threshold at which the oscillator transitions from active oscillations to damping is defined by the variable δ in the definition of μ_{th}. This threshold can be expressed in function of the input signal amplitude, it corresponds to the root mean square (RMS) value of the input signal F:

$$\delta = \frac{A}{\sqrt{2}} \tag{7.18}$$

The excitatory behaviour transitions towards a damping behaviour occurs at $A = \sqrt{2}\delta$, this corresponds to the pitchfork bifurcation point.

Analysis of Hopf oscillator output to an input composed of several sinusoid functions

This section demonstrates how oscillators adapt to composite signals consisting of multiple sinusoids, as well as their ability to separate two distinct signals. It also compares the results

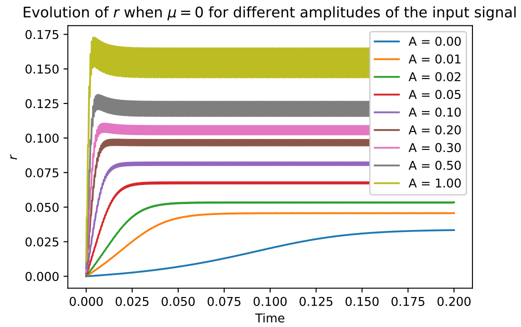


Figure 7.10: The evolution of the radius r of an oscillator for different signals tuned at its central frequency with an adaptable bifurcation parameter.

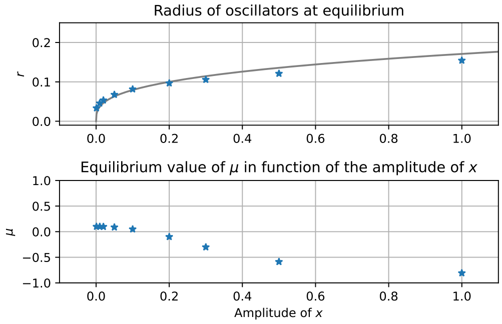


Figure 7.11: Diverging tendencies from the cube root relationship between the input signal amplitude (A) and the oscillator radius (r) with μ_{max} fixed to 0.1.

with those obtained from SincNet filters.

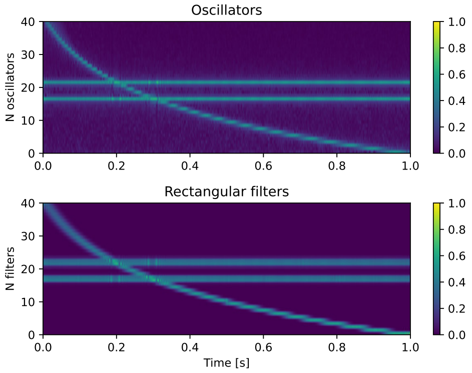


Figure 7.12: Comparison of the output of SincNet filters and oscillators for composite signals of multiple sinusoids.

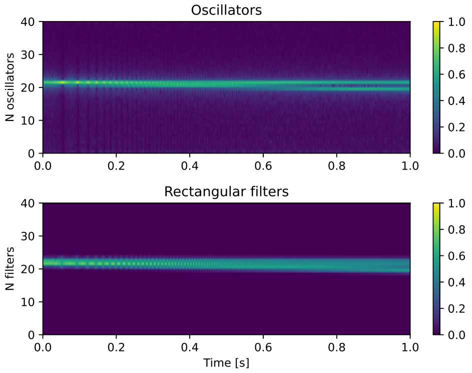


Figure 7.13: Comparison of frequency separation between Hopf oscillators and SincNet filters.

In figures 7.12 and 7.13 we observe interaction between different signals in the contexts of Hopf oscillators and SincNet filters. Figure 7.12 illustrates the oscillator's response to a signal when sweeping its frequency from 0 to $fs/2$ combined with two signals at a specific frequency. This experiment demonstrates the oscillator's response across the considered frequency range, as well as its activation and deactivation capabilities of each oscillator when a signal appears and vanishes at all frequencies. Furthermore, the crossing points between signals reveal how the system responds to an accumulation of multiple signals. Notably, we observe through the colour intensity contrasts in Figures 7.12 and 7.13 that filters multiply the amplitude by 2 while oscillators multiply by $\sqrt[3]{2}$ due to their inherent cube root relationship between the signal amplitude and the oscillator's radius. The places where signals are superimposed correspond to a yellow colour, whereas the rest is around the middle value for the rest of the graph in filter responses and inbetween the middle colour and yellow for the oscillator response. Furthermore the type of response between filters and oscillators differ, oscillators have the tendency to keep a given inertia through the active amplification mechanism.

Figure 7.13 represents one signal at a fixed frequency, while another signal has a sweeping frequency starting from the same point and gradually increasing the frequency. This experiment aims to compare the frequency separation capacity of oscillators with that of SincNet filters. This experiment yields distinct results, the two signals become distinguishable around 0.65 seconds for the oscillators and 0.9 seconds for the SincNet filters. Generally, the oscillators better differentiate closely related signals than SincNet filters with the same amount of filters and equally distributed. However, the oscillator output contains more noisy additions inherent to its inner mechanism.

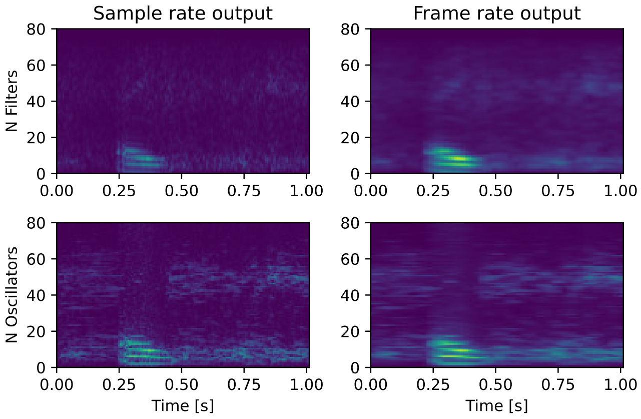


Figure 7.14: Comparison of the Hopf oscillator output wit mel-distributed SincNet filter output for a small utterance: 'eight'.

Analysis of Hopf oscillator output to an speech signal input

More than a simple sum of sinusoids, oscillators can take speech input signals as input. One interesting point in the system output performance analysis is the comparison between the discrete Fourier transform (DFT), the oscillator output and the spectrogram of the waveform. The DFT computation is defined by:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-i2\pi \frac{kn}{N}} \tag{7.19}$$

Where k represents the filter number and n the sample number, N corresponds to the total number of samples.

To be comparable, the DFT and oscillator output should be computed with frames of the same size. Therefore, we take the RMS value of the oscillator output for the different oscillators and we configure the oscillator central frequencies with a linear spacing.

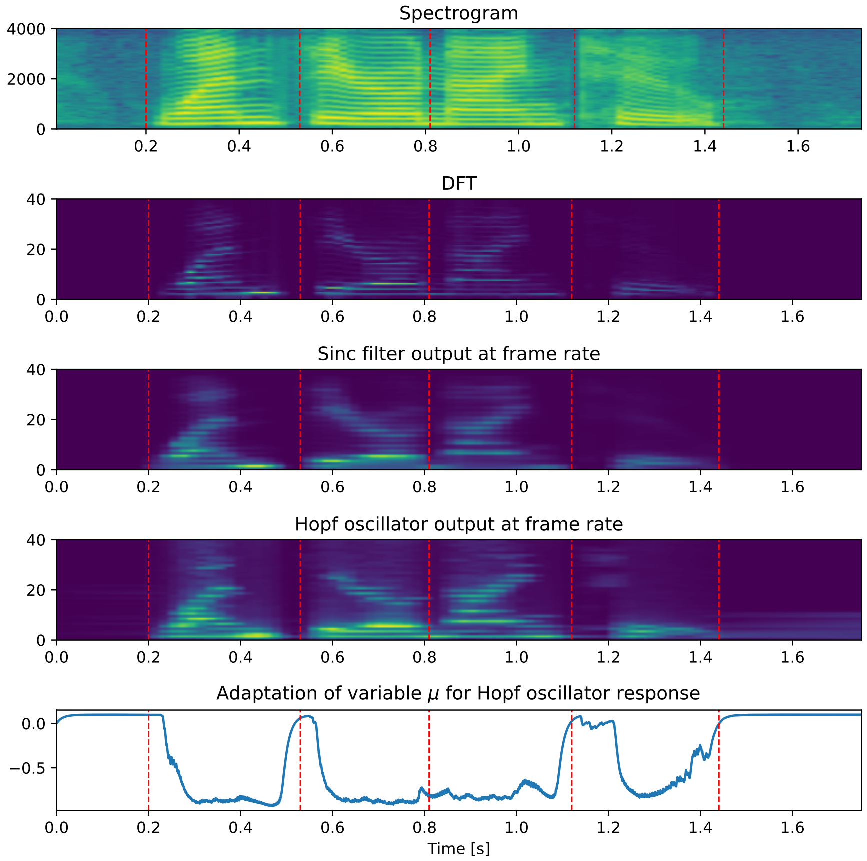


Figure 7.15: Comparison of output of linearly distributed filters with a DFT computation of a four digit utterance: 'one, zero, nine, two', .

Figure 7.14 shows the output of SincNet filters and Hopf oscillators output both at the sample rate and frame rate. Figure 7.15 compares the frame rate values on a linear scale with the DFT computation values. The different signals show a clear resemblance, although some differences can be noticed: the cube root relationship between the amplitude of the signal and the output can be seen by comparing more noisy parts of the signal such as between 0.7 and 0.8 seconds. When no signal is present, the oscillators start to oscillate actively, this phenomenon can be observed between 0.25 and 0.5 seconds for example. The system also tends to keep the activated oscillators oscillating rather than activating all the oscillators directly.

Some inherent differences between SincNet and Hopf oscillators reside in the phase estimation. SincNet filters are iterating over samples without changing the phase of the filter and capture all possible activity at a given frequency, while the oscillators synchronise to get in phase with the input signal, some adaptation time is needed when changing phase in the signal.

Figure 7.16 shows how the bifurcation parameter μ adapts itself when the amplitude of the signal is reduced. In this example, five digits are spoken and the middle one has an amplitude

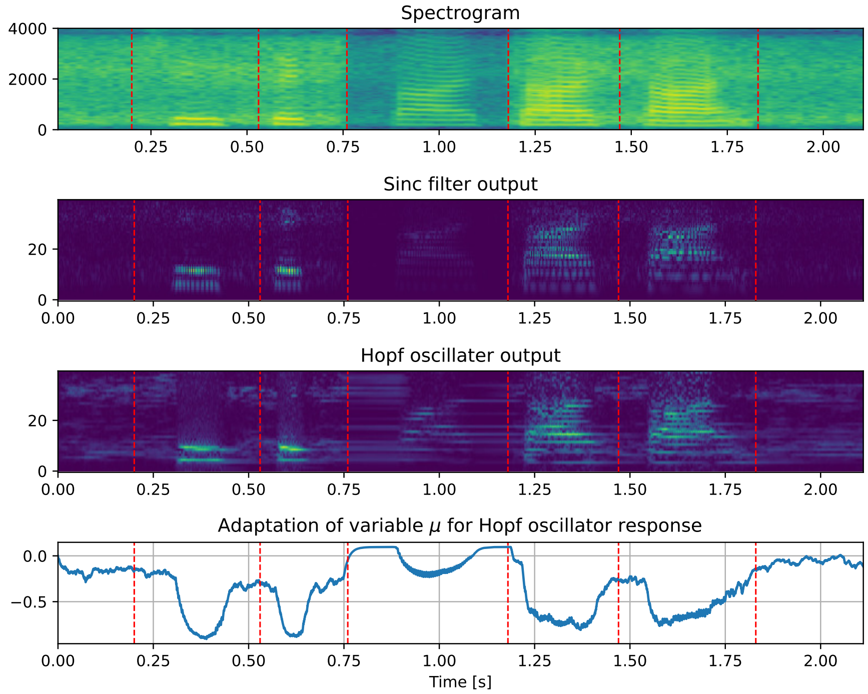


Figure 7.16: Adaptation of μ parameter to a speech input in a noisy environment for a five digit utterance: 'two, six, five, five, five'.

divided by 10. When comparing the output of SincNet filters and the output of the Hopf oscillators, we clearly see the amplification of the low amplitude speech signal. On the μ parameter plot, we also see that for the third digit, the signal is less damped than the 4 other digits. Although the middle digit has an amplitude similar to the surrounding noise, the speech signal better traced than the noise.

7.4 Integration in ASR

7.4.1 Challenges in bigger scale network

The primary challenge in using a recurrent neural network with a physiologically based model lies in managing time consumption during large-scale training. The cochlea's functioning is not optimized for parallel processing, whereas ASR models are designed to be parallelized, where data can be efficiently processed. To overcome this limitation and enable efficient training, generalizations and simplifications must be made.

As a first step towards simplification, we have ignored the coupling component in our Hopf oscillator module implementation, according to the conclusions of section 4.5. Another trade-off arises from choosing the time constant for the active amplification loop in the Hopf oscillator module. With a sampling rate of 16 kHz, adopting this as the adaptation rate provides

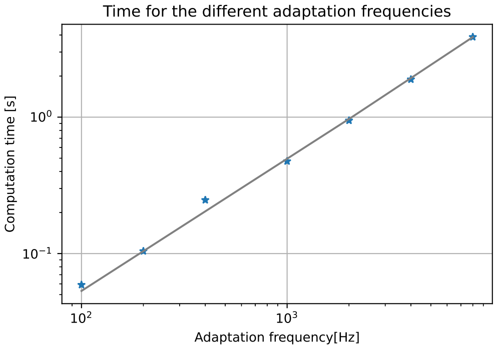


Figure 7.17: Computation time needed for different values of the adaptation frequency going from 8kHz to 100Hz.

optimal precision, but at the cost of computational intensity. Choosing a lower adaptation rate presents an alternative, allowing the recurrent step computation to be applied to a larger number of samples. However, having a lower adaptation rate introduces signal distortion, making it essential to select reasonable frequencies. The frame rate frequency serves as a priori indicator for determining the lower bound of suitable values.

7.4.2 Adapting time constant of active amplification loop

The adaptation of the Hopf module update step implies adapting the mathematical formulation to be mathematically consistent with the original Hopf oscillator expression. To achieve this consistency, modifications are made to the differential equation update step. Specifically:

- The mean value of samples is used instead of individual values for each sample of μ, θ and r. This approach helps ensure that the model captures overall trends in the data rather than focusing on minor variations.
- The initialization of θ is defined by a linear function depending on the central frequency of the oscillator. In contrast, the initialization of μ and r are constant values for the entire vector.
- The value of p in the computation of μ is adapted according to the number of samples N present in each time step. :

  $$p_N = p_1^N \tag{7.20}$$

  where N corresponds to the amount of samples in one time step Δt. N is a parameter that is set at the beginning of an experiment. In this equation p_N represents the value p for a time step of N samples and p_1 represents the value p for a time step of 1 sample.

  A resulting example waveform is shown in Figure 7.18. When the adaptation rate is lowered, the signal becomes distorted. However, visually, the main structures of the waveform remain. For a frame rate of $f = 100Hz$, which corresponds to a adaptation rate equal to the frame rate, the distortion becomes particularly severe.

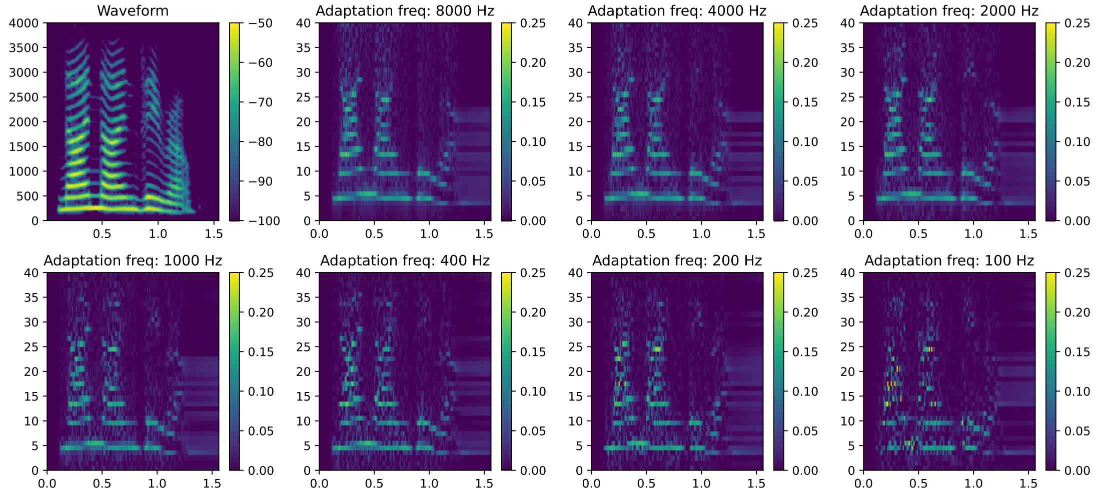


Figure 7.18: Illustration of the oscillator output for different adaptation frequencies for the utterance: 'nine, nine, zero'.

The computational efficiency of the model exhibits a linear relationship with respect to the update frequency, as illustrated by Figure 7.17. This relationship highlights the importance of balancing the trade-off between update frequency and computation time.

The impact of the signal distortion on the performance can be further investigated by using different adaptation frequencies in an ASR task. The PER metric provides a quantitative assessment of the model's performance, allowing to identify the most suitable balance between update frequency and computation time.

7.4.3 Method

The experiments in the following sections are done using the pytorch-kaldi framework, similarly to chapter 5. We used the network structure of SincNet in which we replaced the sinc filter module by the Hopf oscillator module. This Hopf oscillator module is then followed by 4 convolutional layers with the kernelsizes set to: 5,5,3 and 3, the maxpooling size is respectively set to 3,3,2 and 2. The training is performed on 24 epochs and the final results computed on a single run.

7.4.4 Integration of Hopf-module in a simple ASR structure

This section examines the capabilities of the Hopf oscillator module in an ASR context. The integration of the Hopf module in a simple ASR system is depicted in Figure 7.19. The proposed architecture uses two distinct training types as presented in section 7.2: the learning through

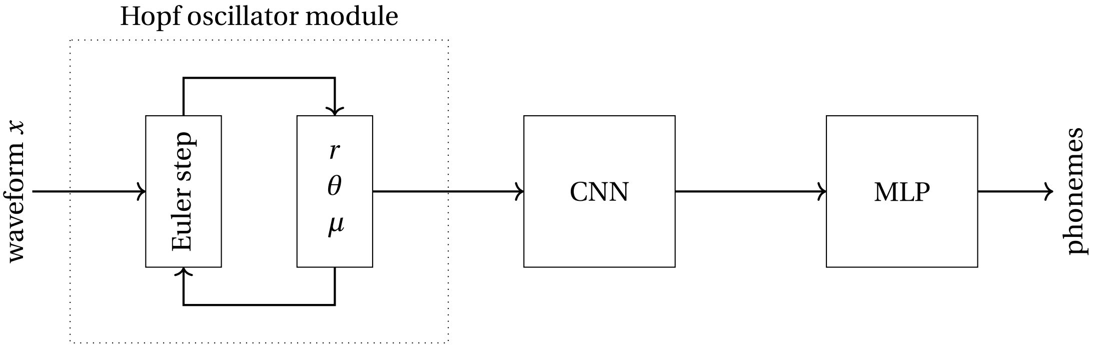


Figure 7.19: Integration of Hopf module in a simple ASR baseline.

gradient descent for the CNN and MLP and the learning through differential equations for the Hopf module.

The first experiment aims to investigate whether the ASR system can produce meaningful performance with a Hopf oscillator feature extractor. Furthermore, it examines which is the optimal adaptation frequency, based on the trade-off between system speed and performance.

Hypothesis

The characterization graphs presented in Section 7.3.3 reveal similarities between SincNet filter outputs, the DFT, and Hopf oscillator outputs. However, due to their active amplification mechanism, noise accumulates in silent parts, which can lead to incorrect phoneme classification in the absence of a proper feedback mechanism. Furthermore, the comparative performance of CARFAC features versus classical MFCC features (as discussed in Section 4.4) showed that cochlear-inspired models perform worse yet remain comparable to classical ASR features in an ASR context. Therefore, we expect to achieve reasonable PER results that demonstrate the system's ability to interpret and extract meaningful features from Hopf oscillator outputs.

Additionally, based on our analysis of the waveform in Figure 7.17, we anticipate that the system's processing time will be significantly longer than that of a standard SincNet filter module. Specifically, while the SincNet filter module processes one utterance in 7.1 ms, the Hopf oscillator feature extractor, at an adaptation frequency of 100 Hz, requires approximately 60 ms. When reducing the Hopf adaptation rate, signal distortion becomes more pronounced.

We hypothesize that at high adaptation rates (16 kHz), an ASR system using a Hopf oscillator feature extractor will yield results comparable to those obtained with fixed SincNet filter outputs. However, at lower adaptation rates, we expect a deterioration in overall performance.

Experiment

For this experiment, we utilize the baseline Hopf oscillator module integrated into a standard ASR system, as depicted in Figure 7.19. We then evaluate the performance of the system and measure the processing time required for each epoch over a period of 24 epochs.

Notably, we perform this evaluation with 40 oscillators and experiment with various adaptation frequencies, specifically: 16kHz, 8kHz, 4kHz, 2kHz, 1kHz, 500Hz, 200Hz, 100Hz.

Results

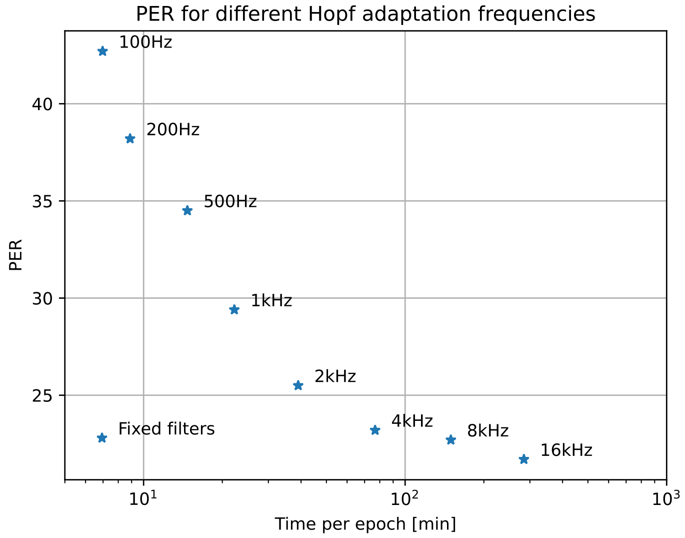


Figure 7.20: PER for different adaptation frequencies from 100Hz to 16kHz and a comparison with fixed filters.

Figure 7.20 summarises the ASR performance achieved for different adaptation frequencies of the active amplification mechanism of the Hopf oscillators. The results suggest that Hopf oscillators generate features that enable effective phoneme recognition within an ASR system as expected.

Furthermore, these results confirm our hypothesis; they indicate a clear trade-off between performance and efficiency, where lower adaptation frequencies result in reduced computational time, but come at the cost of increased WER. Notably, experiments with different frequency settings demonstrate that:

- A 100 Hz adaptation rate offers similar time efficiency to fixed filters, but the performance is around 43% PER.
- With an adaptation frequency between 4 and 16 kHz, the performance capacities of ASR

models are similar to the performance obtained with fixed filters. The time per epoch however is multiplied by a factor between 10 and 50.

These findings suggest that using lower adaptation frequencies for optimization purposes can provide significant computational savings, while sacrificing some accuracy. Conversely, high-frequency experiments are better suited for testing the robustness and capabilities of ASR model structures.

7.4.5 Impact of adding recurrence on the CNN frontend

The auditory system features several feedback connections within its pathway, with the olivocochlear pathway from the olivary complex to the cochlea being one of the most well-known (as described in Section 2.1.4). To adapt the system for the implementation of this efferent pathway, the CNN must be modified to process smaller input chunks in a recurrent manner, which implies adding an adaptation rate at the CNN level, which we will refer to as CNN adaptation rate in this manuscript.

The Hopf features should be transferred to the CNN in a chunk-by-chunk fashion, requiring the convolutional part of the network to handle smaller input signals. Additionally, these chunks must be concatenated before being transmitted to the MLP. These structural modifications are likely to impact ASR performance, and the goal of this experiment is to investigate the effects of these changes on both PER and efficiency.

Hypothesis

The structural modifications will reorganize the processing flow of the signal through the CNN, breaking down the operation of passing one signal into N passes in series of smaller chunks through the same network. We hypothesize that this decomposition will have a direct and significant impact on computational time.

In terms of performance, we do not a priori expect discernible difference in results between the original system and the modified system, with neither exhibiting better nor worse performance in terms of PER.

Experiment

This experiment is a transitional experiment in order to integrate the efferent path and verify the behaviour of the ASR capacities when structurally changing the CNN. Based on the time efficiency of the experiment in section 7.4.4, a reasonable choice to perform this experiment is to use the Hopf adaptation rate of 100Hz, 200Hz, and 500Hz. In the initial experiment, those configurations took less than 20 minutes per epoch.

The speech signals are composed out of 3200 samples, which correspond to 200 ms or a CNN adaptation rate of 5 Hz. The CNN adaptation rate can make sense up to the frame rate, which corresponds to 100 Hz. However, with downsampling in the CNN system, we are limited in terms of precision. Table 7.1 summarizes the input and output sizes for different CNN adaptation rates.

CNN adaptation rate	Dim at input of CNN	Dim at output of CNN
100Hz	160	2
50Hz	320	6
20Hz	800	20
10Hz	1600	42
5Hz	3200	86

Table 7.1: Input and output size of the CNN for different CNN adaptation rates.

A trade-off between having the CNN adaptation rate small enough to allow feedback to influence the input and large enough to be still able to have significant information to transfer to the MLP after downsampling through the CNN.

For 100 Hz, for example, the output dimension in the time domain is 2, which means that with normalization, the output loses all information. In this experiment, we try out the following CNN adaptation rates: 50Hz, 20Hz, 10Hz, 5Hz, with 5 Hz corresponding to the results obtained in section 7.4.4.

Results

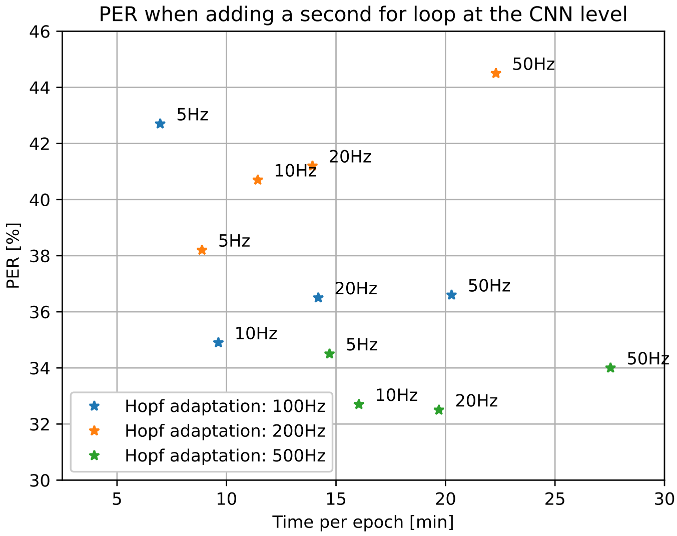


Figure 7.21: PER vs time consumption when introducing a for loop with CNN adaptation rates of 5Hz, 10Hz, 20Hz and 50Hz around the CNN.

The performance and time taken for the different experiments are shown in Figure 7.21. Concerning efficiency, increasing the CNN adaptation rate increases the time consumption which demonstrates our hypothesis. Concerning performance, adding a CNN adaptation rate degrades the performance for some experiments and improves it for others. The general trend, however, is that except for the 5Hz to 10Hz transition, the more iterations we have, the more the performance will be degraded.

7.4.6 Adding the efferent path feedback loop

The most important feedback loop in the auditory path is the olivocochlear feedback as discussed in section 2.1.4. This feedback is divided into two parts: one from the MOC to the OHCs and one from the LOC to the synapses of the IHCs.

When translating the efferent pathway to a computational area, a first implementation can be given by creating a feedback loop from the output of the CNN through a transposed convolution network (TCNN) as depicted in Figure 7.22. The feedback can then be combined through a linear module with the next incoming signal portions.

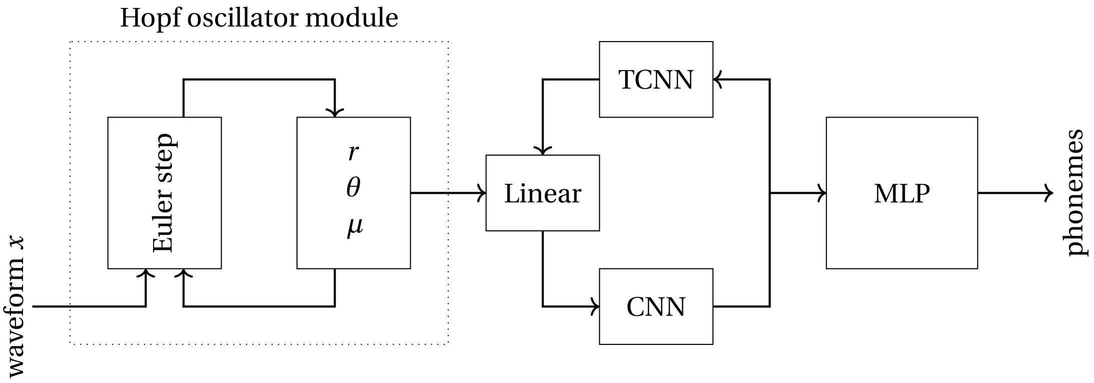


Figure 7.22: Attention feedback loop

Hypothesis

Since the efferent pathway plays an important role in the auditory path, we hypothesize that adding a similar feedback loop in the ASR pipeline would have a positive effect on the performance of the ASR. However, from a computational efficiency perspective, the system will be less efficient than when we use the baseline ASR configuration described in section 7.4.4.

Experiment

For this experiment, we select 50Hz as the CNN adaptation rate, since this allows the feedback signal to have an update every 200ms. We perform the experiments with different Hopf adaptation frequencies: 100Hz, 200Hz, 500Hz, 1kHz, 2kHz, 4kHz. This allows us to analyze the performance-efficiency trade-off and compare it to the results of section 7.4.4.

The combination of the feedback signal with the Hopf oscillator output is done through a linear module, which requires a proper initialization. If the module starts from a random initialization, it does not converge. Therefore, we start the initialization by concatenating an identity matrix for the Hopf oscillator output and a matrix of zeros for the feedback part. This way, the system starts with no influence of the feedback mechanism.

Results

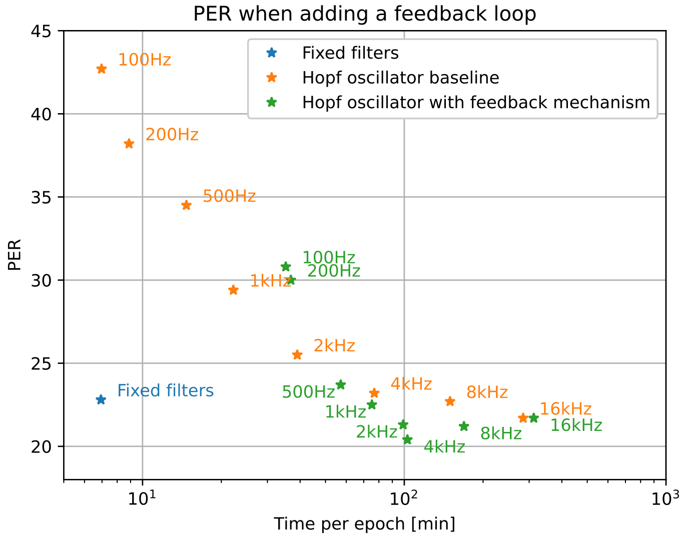


Figure 7.23: PER results when adding a large feedback loop.

The results are summarized in Figure 7.23. The addition of a large feedback loop is, on the one hand, computationally more demanding, but on the other hand, it brings a consistent performance improvement over all the tested configurations as we hypothesised.

For Hopf adaptation frequencies below 500Hz, the baseline implementation provides a better efficiency-performance trade-off. For higher Hopf adaptation frequencies, however, the performance beats this trade-off and achieves even better results than the baseline at 16kHz.

This result indicates that the Hopf mechanism on its own does provide an output that requires an additional feedback mechanism, in the same way as the cochlea is better tuned with the efferent path.

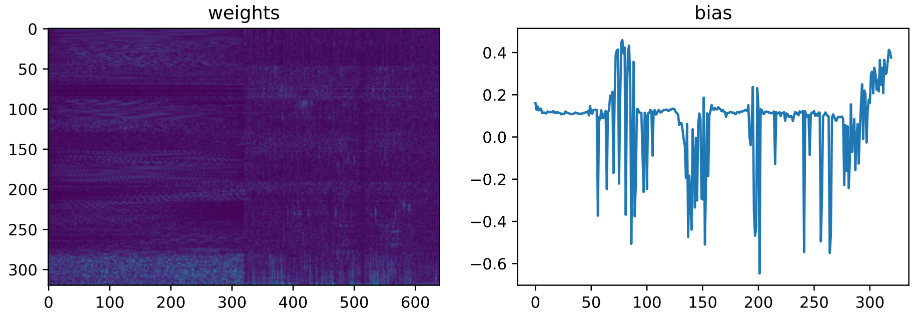


Figure 7.24: Weight and bias of the linear combination module after training.

Furthermore, the weight matrix of the linear module that combines the Hopf oscillator output with the feedback signal is shown in Figure 7.24. A clear separation is visible: the left side relates to the Hopf oscillator output and is initialized as an identity matrix. The right side relates to the feedback component, which is initialized as a matrix of zeros, implying that at the beginning of the training no feedback is taken into account.

As illustrated in Figure 7.24, the feedback part of the matrix plays an important role in determining the weights. The combined increase in performance indicates that this feedback loop is crucial for the ASR system to deal effectively with outputs from a Hopf bifurcation mechanism. This triggers an interesting question towards the neuroscientific field: what would be the auditory performance without the efferent pathway in terms of speech recognition?

7.4.7 A more complex implementation

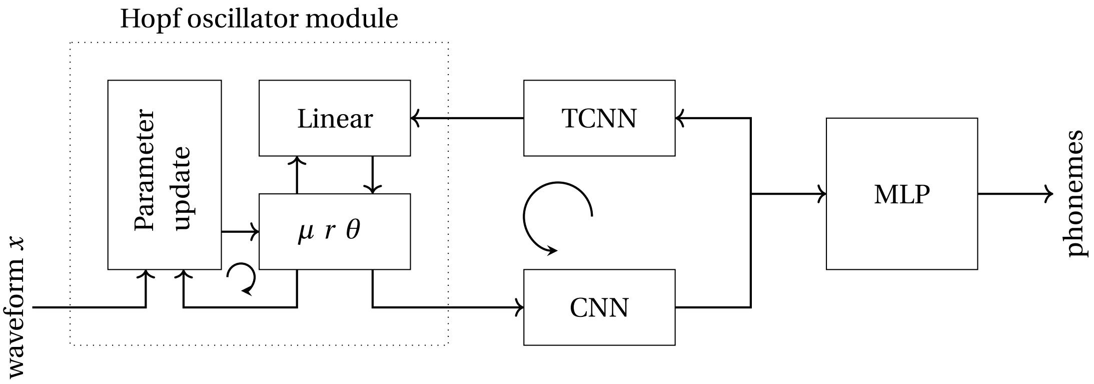


Figure 7.25: Attention feedback loop

Another possible implementation of the efferent path would be to directly integrate the feedback into the Hopf module. This implementation better integrates the notion of a direct impact on the OHC, modeling the connection between the MOC and OHC. This approach is computationally more complex to implement, as the gradients must pass through the μ, r and θ variables, which are changing at a faster adaptation rate than the large feedback. This causes difficulties in backpropagation due to inplace operations during the forward pass. Moreover, the Hopf oscillator equations require strict regulations to prevent the system from diverging. The use of a random parameter that trains within a neural network system is though to handle this differential equation system. Due to the implementation complexity and the thorough stability issues, we were unable to conduct extensive experiments with this implementation. However, this model structure can be an interesting starting point for further research in speech processing with the olivocochlear feedback implementation.

7.5 Noise addition

The fundamental difference in how Hopf oscillators and convolutional filters handle incoming signals stems from the presence of an active amplification mechanism and cube root compression within the Hopf oscillator. This characteristic, typical of biological processes found in the human ear, suggests that it may possess valuable properties in terms of noise robustness (Shougat et al., 2021, 2023).

In the realm of computational signal processing, this phenomenon presents a compelling research direction. By exploring how these models perform under varying conditions, we can gain a deeper understanding of their strengths and weaknesses. Specifically, we aim to investigate the performance differences between Hopf oscillators and convolutional filters when trained on noisy data versus clean data.

To achieve this goal, we propose two complementary experiments:

- **Clean-to-Noise Transfer** : We use a model trained on clean speech and then evaluate its performance under increasingly noisy conditions.
- **Training from Scratch** : We train a new model using a noisy dataset and evaluate this model under the same noise conditions.

By examining the performance of these models in both scenarios, we can gain insights into their ability to generalize from clean data to noisy environments.

Hypothesis

We hypothesise that the integration of the Hopf oscillator mechanism alongside the efferent path could confer several advantages over a traditional fixed filterbank approach. The incorporation of global feedback and an active amplification mechanism may enable the system to better adapt to complex, noisy environments.

Furthermore, we hypothesise that the performance of oscillators will exhibit reduced degradation compared to filters when operating within a feedback loop, particularly when only testing on noisy data with a model trained on clean data. This is likely due to the oscillator's ability to actively amplify and process signals, allowing it to better cope with the challenges presented by noise.

In contrast, we hypothesise that the performance of the ASR system, when trained from scratch, will exhibit relatively little difference between utilizing a Hopf oscillator frontend versus a convolutional filter frontend. This is because both architectures are expected to undergo significant adaptation and learning during training, allowing them to effectively process and generalize from noisy data.

Experiment

In this experiment, we introduce noise on the fly using the 'QUT-Noise' dataset (presented in section 2.3.4) on the TIMIT dataset. For each speech portion, a random noise signal is added in accordance with the chosen Signal-to-Noise Ratio SNR. To capture the main trends, we conduct all experiments with three different SNR rates: 0 dB, 10dB and 20 dB. This experiment is performed on the model with an efferent path adaptation rate of 50 Hz and Hopf the adaptation rates of 100Hz, 200Hz, 500Hz, 1kHz, 2kHz, 4kHz.

Results

The results are presented in Tables 7.2 and 7.3. Notably, performance degrades across all experiments as the SNR decreases. In the clean-to-noise transfer experiment (Table 7.2), the results indicate that Hopf oscillators exhibit reduced robustness to noise additions compared to convolutional filters when trained on a clean dataset. This suggests that while Hopf oscillators may mathematically capture some aspects of cochlear physiology, they do not necessarily provide superior noise robustness. In contrast, the second experiment, where all models are trained from scratch (Table 7.3), reveals that the degradation of Hopf filters is comparable to that observed with classical filters. This outcome implies that, in terms of noise robustness, Hopf oscillators do not exhibit an advantage over convolutional filters, which contradicts our hypothesis.

	clean	20 dB	10 dB	0 dB
sinc filters	22.8	26.0	36.1	54.6
100Hz	30.8	36.5	51.4	68.3
200Hz	30.0	36.4	54.2	74.1
500Hz	23.7	30.8	49.4	72.4
1kHz	22.5	29.0	46.8	71.7
2kHz	21.3	26.8	45.1	68.8
4kHz	20.4	23.9	38.1	64.9

Table 7.2: PER for the different SNR levels on the TIMIT dataset with a model trained on clean speech.

	clean	20 dB	10 dB	0 dB
sinc filters	22.8	22.7	23.4	29.3
100Hz	30.8	32.7	35.5	42.6
200Hz	30.0	31.7	31.9	39.6
500Hz	23.7	24.4	27.6	33.6
1kHz	22.5	22.8	25.5	30.2
2kHz	21.3	21.8	24.2	28.9
4kHz	20.4	20.7	22.8	28.7

Table 7.3: PER for the different SNR levels on the TIMIT dataset with a model trained on noisy speech

7.6 Hardware implementations of Hopf oscillators

Research has explored the hardware implementation of Hopf oscillators with adaptive frequency mechanisms, as proposed in the literature (X. Li et al., 2021). These studies have examined the feasibility of leveraging phase-locked loop (PLL) principles to develop an adaptive oscillator mechanism.

In this context, a PLL is an electronic circuit that utilizes feedback to adjust its output phase to match the input frequency. By replicating this behavior, researchers aim to design Hopf oscillators capable of adapting to changing environmental conditions.

The Hopf oscillator reservoir proposed by (Shougat et al., 2021, 2023) also demonstrated a computationally efficient electronic implementation. This opens promising research directions in the field of electronics to further investigate a feasible implementation of cochlear function, potentially drawing an interesting parallel to this thesis.

7.7 Conclusion

This chapter presents the integration and implementation of a module inspired by the Hopf oscillator mechanism into an ASR system. The Hopf oscillator, mathematically describing the OHC-IHC interaction, enables the human auditory system to process a wide range of sound amplitudes through cube-root compression and active amplification.

Upon incorporation into the ASR system, the module introduces a dual adaptation mechanism: one driven by differential equations to dynamically adjust the parameters of the Hopf module, and another governed by an autograd system to optimize the weights of the ASR's various components.

Furthermore, this chapter incorporates larger auditory pathway feedback mechanisms into the ASR system, which parallels the feedback process of the auditory brain. The autograd mechanism adjusts synaptic weights to enhance the tonotopic mapping of incoming sounds, while a larger CNN-TCNN architecture models the olivocochlear feedback loop, directly influencing the Hopf module's output. The results demonstrate that this feedback mechanism significantly improves experimental performance, highlighting its necessity for adapting Hopf module signals to phoneme mapping. In physiological terms, the olivocochlear feedback loop is a well-documented feedback pathway in the auditory system essential for human hearing.

Moreover, previous studies suggest that the feedback mechanisms implied in the ear enhance noise robustness for human hearing. However, our experiments reveal that this capacity is not observed compared to convolutional filters in the context of ASR. Nevertheless, we do not exclude the possibility that other implementations could lead to more robust behaviour, such as in self-supervised model contexts.

8 Conclusion

This thesis lies at the intersection of the ASR field and the physiological understanding of the cochlea. In ASR, self-supervised pre-trained models are the current state-of-the-art. These models are pretrained on large amounts of unlabeled data and able to be fine-tuned on labeled datasets. The latest models use transformer-based modules, which achieve state-of-the-art performance, but are computationally more expensive compared to the previous vanilla ASR models. The cochlea can initially be understood to work as a filterbank. However, recent studies suggest that the cochlea functions more like an array of active amplification oscillators, driven by a local feedback loop between OHCs and IHCs. The mathematical model that best approaches this interaction is the Hopf bifurcation model. The bifurcation enables the model to switch from a damping mode in loud environments to an active amplification mode in silent environments.

Modularity is a key concept in the combination of different modules. In the context of this thesis, the modularity concept is particularly interesting for the integration of cochlear models within ASR systems. Additionally, we investigated the modularity concept for noisy speech in a conformer-based ASR system. In this study, we implemented a fixed and learned routing mechanism to route speech in different noise environments and showed that using a modular network enhanced the overall performance as well as the learning curve.

Building on the modularity concept, we undertook two studies using trainable filters of SincNet into ASR systems. An initial study was conducted on a small ASR system, highlighting that the filters tend to learn both narrowband and wideband filters when trained within an ASR system. Narrowband filters self-organize into a mel-distributed filterbank of 30 to 40 filters. Wideband filters capture larger frequency range information, which in physiology are found in higher auditory path nuclei. In a second study, we investigated the trainable filters within a self-supervised model. This study confirms that a trainable filterbank tends to learn about 40 filters. This study also shows that wideband filters are precluded to appear by the self-supervised transformer-based model.

The Hopf oscillator is the state-of-the-art mathematical model that best captures the intricacies of the cochlea. In the first chapter, we presented the Hopf model, explaining the bifurcation mechanism and demonstrating its ability to process raw speech. This prior work lays the foundation for a second chapter, presenting our Hopf module implementation and integration into an ASR frontend. Moreover, we integrated a larger feedback loop, which models the efferent pathway. This efferent path shows significant performance improvement over the initial Hopf integration. Further, based on the promising noise robustness hypothesis, we performed a series of experiments on noisy data. However, in our implementation, these noise robustness implications did not surpass traditional filter capacities.

Generally, this thesis has investigated modern theories of the cochlea in the context of machine learning models. It explores the use of machine learning models under the assumption that they approximate the human auditory pathway when trained on ASR tasks. This gives interesting insights when combined with more physiologically plausible models, forcing the model to better adapt to more physiologically plausible inputs. The results obtained with the efferent path integration shows on the one hand that physiological inspirations can lead to interesting improvements for the ASR field. On the other hand, using physiologically plausible models can give insights to neuroscientists. The results obtained in deep learning applications raise interesting questions about how the physiological system handles different situations, which was the main purpose of this thesis.

8.1 Further research and recommendations

This thesis acts as a tool for the two communities to mutually inspire each other and contains several results which pose interesting research questions in the physiological field.

For further research, we suggest investigating new implementation solutions that deal with the computational limitations, which can allow actual combinations of state-of-the-art or physiologically plausible ASR models with Hopf-based oscillator frontends. For example, creating a pretrained model for the CNN encoder, which can then be integrated into a larger transformer-based model, may yield interesting insights. Another research direction is to combine physiologically inspired models of the auditory path such as spiking neural networks with the Hopf oscillator frontend.

Hardware implementations of the Hopf oscillators have shown interesting results on sound classification. Combining those electronic Hopf reservoirs with other physiologically plausible circuits could lead to an interesting parallel of this thesis in the electronics field.

Furthermore, the midbrain also employs lateral inhibition to sharpen neural responses by suppressing the activity of neighbouring neurons. This mechanism enhances contrast and facilitates feature discrimination. The maxpooling function partially reflects this idea by selecting only the strongest activations for subsequent layers. However, a more physiologically plausible approach could involve using Mexican hat filters in the convolutional layers following the cochlear stage. When combined with a feedback mechanism, this implementation would more closely resemble the biological auditory pathway, contributing to a more realistic ASR model.

A An appendix

A.1 Active force: a set of two differential equations

We start from the harmonic oscillator equation combined with an active amplification force:

$$m\ddot{x} = -\lambda\dot{x} - kx + F_a + F \tag{A.1}$$

If we consider the inertial effects as negligeable ($m\ddot{x} = 0$), the stiffness as nonlinear ($k = k(x)$) then this harmonic oscillator equation becomes:

$$\lambda\dot{x} = -k(x)x + F_a + F \tag{A.2}$$

To obtain oscillations, the non-linear stiffness should display a regime of negative elasticity $k(x) = k - C + Bx^2$ where k is the bare stiffness and C a control parameter characterizing a reduction of stiffness. When $C > k$ the system actively amplifies the oscillations for small displacements x. The active force f_a evolves as a first-order differential equation that generates a restoring force when the system is displaced which relaxes with a time-constant τ and its own stiffness $\bar{k}$.

$$\begin{cases} \lambda\dot{x} = -(k - C + Bx^2)x + F_a + F \\ \tau\dot{F}_a = -F_a - \bar{k}x \end{cases} \tag{A.3}$$

A.2 Equations for simulation of Hopf oscillator

A.2.1 Single oscillator with adaptive frequency

This appendix explains more in detail how the simulation equations of chapter 4.3 the equations are derived from Biswas et al., 2020.

The canonical model of the Hopf oscillator without any external input is given by:

$$\dot{z} = z(\mu + i\omega + \beta_1 |z|^2) \tag{A.4}$$

Further developing this expression in polar form gives:

$$\dot{z} = \dot{r}e^{i\theta} + ire^{i\theta}\dot{\theta} = re^{i\theta}(\mu + i\omega + \beta_1 r^2) \tag{A.5}$$

After separating the real and imaginary parts, the polar coordinates are obtained:

$$\begin{aligned} \dot{r} &= r(\mu + \beta_1 r^2) && \text{(A.6)} \\ \dot{\theta} &= \omega && \text{(A.7)} \end{aligned}$$

Similarly, the Hopf oscillator equation can be expressed with cartesian coordinates:

$$\begin{aligned} \dot{x} &= x(\mu - \sqrt{x^2 + y^2}^{\,2}) - y\omega && \text{(A.8)} \\ \dot{y} &= y(\mu - \sqrt{x^2 + y^2}^{\,2}) + x\omega && \text{(A.9)} \end{aligned}$$

A Hopf oscillator influenced by a real sinusoidal input ($F(t) = Fcos(\omega_0 t + \phi)$) signal can adapt its natural frequency to the frequency of the input signal if it follows the following dynamics (Righetti et al., 2005). The canonical model of the Hopf oscillator with an external input and the oscillator adaptation are given by:

$$\begin{aligned} \dot{z} &= z(\mu + i\omega + \beta_1 |z|^2) + F(t) && \text{(A.10)} \\ \dot{\omega} &= -F(t)\sin\theta && \text{(A.11)} \end{aligned}$$

The polar coordinates are given by:

$$\dot{r} = r(\mu - r^2) + F(t)\cos(\theta) \tag{A.12}$$

$$\dot{\Phi} = \omega + \frac{F(t)}{r}\sin(\theta) \tag{A.13}$$

$$\dot{\omega} = -F(t)\sin\theta \tag{A.14}$$

For a complex input($F(t) = Fe^{i(\omega_0 t + \phi)}$), the frequency adaptation function becomes:

$$\dot{\omega} = -[real(F(t))\sin(\theta) - imag(F(t))\cos(\theta)] \tag{A.15}$$

$$= -[F\cos(\omega_0 t + \phi)\sin(\theta) - F\sin(\omega_0 t + \phi)\cos(\theta)] \tag{A.16}$$

$$= -F\sin(\theta - \omega_0 t - \phi) \tag{A.17}$$

A.2.2 A series of Hopf oscillators

To model a series of Hopf oscillators, we need to take into account the impact of the interaction between the different oscillators. Therefore the interaction between the oscillators is modelled through complex coefficients with Hermitian symmetry. For two oscillators i and j with the same frequency (ω), the Hopf oscillator equation of oscillators becomes:

$$\dot{z}_i = z_i(\mu + i\omega + \beta|z_i|^2) + Wz_j \tag{A.18}$$

$$\dot{z}_j = z_j(\mu + i\omega + \beta|z_j|^2) + W^* z_i \tag{A.19}$$

Where $W = A_{ij}e^{i\theta_{ij}}$ and $W^* = A_{ji}e^{i\theta_{ji}}$ with $A_{ij} = A_{ji}$ and $\theta_{ij} = -\theta_{ji}$.

Oscillators can however have different frequencies, the angular coupling should therefore be normalized: $W = A_{ij}e^{i\frac{\theta_{ij}}{\omega_j}}$ and $W^* = A_{ji}e^{\frac{\theta_{ji}}{\omega_i}}$. The signal from the other oscillator also undergoes a coupling frequency adaptation : $z_j \rightarrow z_j^{\frac{\omega_i}{\omega_j}}$:

$$\dot{z}_i = z_i(\mu + i\omega_i + \beta|z_i|^2) + A_{ij}e^{i\frac{\theta_{ij}}{\omega_j}} z_j^{\frac{\omega_i}{\omega_j}} \tag{A.20}$$

$$\dot{z}_j = z_j(\mu + i\omega_i + \beta|z_j|^2) + A_{ji}e^{i\frac{\theta_{ji}}{\omega_i}} z_j^{\frac{\omega_j}{\omega_i}} \tag{A.21}$$

$$\tag{A.22}$$

Extending this model to a series of N oscillators results in the canonical model given by:

$$\dot{z}_i = \underbrace{z_i(\mu + i\omega_i + \beta|z_i|^2)}_{\text{oscillator } i} + \underbrace{\sum_{j,i\neq j}^{N} A_{ij} e^{i\frac{\theta_{ij}}{\omega_j}} z_i^{\frac{\omega_i}{\omega_j}}}_{\text{coupling with other osc.}} + \overbrace{F(t)}^{\text{ext. signal}} \tag{A.23}$$

Bibliography

Agrawal, P., & Ganapathy, S. (2019). Unsupervised raw waveform representation learning for asr. *INTERSPEECH*, 3451–3455.

Allen, J. B. (1980). Cochlear micromechanics—a physical model of transduction. *The Journal of the Acoustical Society of America*, *68*(6), 1660–1670.

Amari, S.-i. (1993). Backpropagation and stochastic gradient descent method. *Neurocomputing*, *5*(4-5), 185–196.

Ammari, H., & Davies, B. (2020). Mimicking the active cochlea with a fluid-coupled array of subwavelength hopf resonators. *Proceedings of the Royal Society A*, *476*(2234), 20190870.

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al. (2016). Deep speech 2: end-to-end speech recognition in English and Mandarin. *International conference on machine learning*, 173–182.

Ansell, A., Ponti, E. M., Korhonen, A., & Vulić, I. (2021). Composable sparse fine-tuning for cross-lingual transfer. *arXiv preprint arXiv:2110.07560*.

Babu, A., Wang, C., Tjandra, A., Lakhotia, K., Xu, Q., Goyal, N., Singh, K., von Platen, P., Saraf, Y., Pino, J., et al. (2021). Xls-r: self-supervised cross-lingual speech representation learning at scale. *arXiv preprint arXiv:2111.09296*.

Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J., & Auli, M. (2022, July). Data2vec: a general framework for self-supervised learning in speech, vision and language. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, & S. Sabato (Eds.), *Proceedings of the 39th international conference on machine learning* (pp. 1298–1312, Vol. 162). PMLR.

Baevski, A., Schneider, S., & Auli, M. (2019). Vq-wav2vec: self-supervised learning of discrete speech representations. *arXiv preprint arXiv:1910.05453*.

Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). Wav2vec 2.0: a framework for self-supervised learning of speech representations. *arXiv preprint arXiv:2006.11477*.

Baevski, A., Zhou, Y., Mohamed, A., & Auli, M. (2020). Wav2vec 2.0: a framework for self-supervised learning of speech representations. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in neural information processing systems* (pp. 12449–12460, Vol. 33). Curran Associates, Inc.

Bak, P., Tang, C., & Wiesenfeld, K. (1988). Self-organized criticality. *Physical review A*, *38*(1), 364.

Balestriero, R., Cosentino, R., Glotin, H., & Baraniuk, R. (2018). Spline filters for end-to-end deep learning. *Proceedings of the international conference on machine learning*, 364–373.

Bedi, G., Carrillo, F., Cecchi, G. A., Slezak, D. F., Sigman, M., Mota, N. B., Ribeiro, S., Javitt, D. C., Copelli, M., & Corcoran, C. M. (2015). Automated analysis of free speech predicts psychosis onset in high-risk youths. *npj Schizophrenia*, *1*(1), 1–7.

Biswas, D., Sooryakiran, P., & Chakravarthy, V. S. (2020). A complex-valued oscillatory neural network for storage and retrieval of multichannel electroencephalogram signals. *bioRxiv*.

Bogert, B. P. (1963). The quefrency alanysis of time series for echoes; cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. *Time series analysis*, 209–243.

Brownell, W. E., Bader, C. R., Bertrand, D., & De Ribaupierre, Y. (1985). Evoked mechanical responses of isolated cochlear outer hair cells. *Science*, *227*(4683), 194–196.

Burget, L., & Heřmanskỳ, H. (2001). Data driven design of filter bank for speech recognition. *International Conference on Text, Speech and Dialogue*, 299–304.

Camalet, S., Duke, T., Jülicher, F., & Prost, J. (2000). Auditory sensitivity provided by self-tuned critical oscillations of hair cells. *Proceedings of the national academy of sciences*, *97*(7), 3183–3188.

Chakraborty, S., Tomsett, R., Raghavendra, R., Harborne, D., Alzantot, M., Cerutti, F., Srivastava, M., Preece, A., Julier, S., Rao, R. M., et al. (2017). Interpretability of deep learning models: a survey of results. *Proceedings of the IEEE smartworld, ubiquitous intelligence & computing, advanced & trusted computed, scalable computing & communications, cloud & big data computing, Internet of people and smart city innovation*, 1–6.

Chan, W., Jaitly, N., Le, Q., & Vinyals, O. (2016). Listen, attend and spell: a neural network for large vocabulary conversational speech recognition. *ICASSP*, 4960–4964. https://doi.org/10.1109/ICASSP.2016.7472621

Collobert, R., Puhrsch, C., & Synnaeve, G. (2016). Wav2letter: an end-to-end convnet-based speech recognition system [Presented at NIPS 2016]. https://doi.org/10.48550/ARXIV.1609.03193

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. *Proceedings of the 25th international conference on Machine learning (ICML)*, 160–167. https://doi.org/10.1145/1390156.1390177

Coppieters de Gibson, L., & Garner, P. N. (2022). Low-level physiological implications of end-to-end learning of speech recognition, 749–753. https://doi.org/10.21437/Interspeech.2022-10093

Cramer, B., Stradmann, Y., Schemmel, J., & Zenke, F. (2020). The Heidelberg Spiking Data Sets for the Systematic Evaluation of Spiking Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 1–14. https://doi.org/10.1109/TNNLS.2020.3044364

Dahl, G., Yu, D., Deng, L., & Acero, A. (2012). Context-dependent pre-trained deep neural networks for large vocabulary speech recognition [2013 IEEE SPS Best Paper Award], *20*(1), 30–42. https://doi.org/10.1109/TASL.2011.2134090

Dahl, G. E., Yu, D., Deng, L., & Acero, A. (2011). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing, 20*(1), 30–42.

Davis, S., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 28*(4), 357–366. https://doi.org/10.1109/TASSP.1980.1163420

De Boer, E. (1983). On active and passive cochlear models—toward a generalized analysis. *The Journal of the Acoustical Society of America, 73*(2), 574–576.

De Boer, E., & De Jongh, H. (1978). On cochlear encoding: potentialities and limitations of the reverse-correlation technique. *The Journal of the Acoustical Society of America, 63*(1), 115–135.

Dean, D., Sridharan, S., Vogt, R., & Mason, M. (2010). The qut-noise-timit corpus for evaluation of voice activity detection algorithms. *Proceedings of the 11th annual conference of the international speech communication association*, 3110–3113.

Dubey, H., Sangwan, A., & Hansen, J. H. (2019). Transfer learning using raw waveform sincnet for robust speaker diarization. *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6296–6300.

Duifhuis, H. (2011). Hopf-bifurcations and van der pol oscillator models of the mammalian cochlea. *AIP Conference Proceedings, 1403*(1), 199–205.

Duke, T. A., & Jülicher, F. (2008). Critical oscillators as active elements in hearing. *Active processes and otoacoustic emissions in hearing*, 63–92.

Garofolo, J. S. (1993). Timit acoustic phonetic continuous speech corpus. *Linguistic Data Consortium, 1993.*

Geisler, C. D. (1976). Mathematical models of the mechanics of the inner ear. In *Auditory system* (pp. 391–415). Springer.

Geisler, C. D. (1986). A model of the effect of outer hair cell motility on cochlear vibrations. *Hearing research, 24*(2), 125–131.

Gianoli, F., Hogan, B., Dilly, É., Risler, T., & Kozlov, A. S. (2022). Fast adaptation of cooperative channels engenders hopf bifurcations in auditory hair cells. *Biophysical Journal, 121*(6), 897–909. https://doi.org/10.1016/j.bpj.2022.02.016

Gianoli, F., Risler, T., & Kozlov, A. S. (2017). Lipid bilayer mediates ion-channel cooperativity in a model of hair-cell mechanotransduction. *Proceedings of the National Academy of Sciences, 114*(51), E11010–E11019.

Gold, T. (1948). Hearing. ii. the physical basis of the action of the cochlea. *Proceedings of the Royal Society of London. Series B-Biological Sciences, 135*(881), 492–498.

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

Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., et al. (2020). Conformer: convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100.*

Hamilton, T. J., Jin, C., Tapson, J., & van Schaik, A. (2007). A 2-d cochlea with hopf oscillators. *Proceedings of the IEEE Biomedical Circuits and Systems Conference*, 91–94. https://doi.org/10.1109/BIOCAS.2007.4463316

Hamilton, T. J., Tapson, J., Jin, C., & Van Schaik, A. (2008). Analogue vlsi implementations of two dimensional, nonlinear, active cochlea models. *Proceedings of the Biomedical Circuits and Systems Conference*, 153–156. https://doi.org/10.1109/BIOCAS.2008.4696897

Hermansky, H. (1990a). Perceptual linear predictive (plp) analysis of speech. *the Journal of the Acoustical Society of America*, *87*(4), 1738–1752.

Hermansky, H. (1990b). Perceptual linear predictive (PLP) analysis of speech. *The Journal of the Acoustical Society of America (JASA)*, *87*(4), 1738–1752. https://doi.org/10.1121/1.399423

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal processing magazine*, *29*(6), 82–97.

Hopf, E. (1942). Abzweigung einer periodischen lösung von einer stationären lösung eines differentialsystems. *Ber. Math.-Phys. Kl Sächs. Akad. Wiss. Leipzig*, *94*, 1–22.

Hu, E. J., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. (2021). LoRA: low-rank adaptation of large language models. *International Conference on Learning Representations*.

Hu, K., Li, B., Sainath, T. N., Zhang, Y., & Beaufays, F. (2023). Mixture-of-expert conformer for streaming multilingual ASR. *arXiv preprint arXiv:2305.15663*.

Hudspeth, A., Jülicher, F., & Martin, P. (2010a). A critique of the critical cochlea: hopf–a bifurcation–is better than none. *Journal of neurophysiology*, *104 3*, 1219–29.

Hudspeth, A. (2008). Making an effort to listen: mechanical amplification in the ear. *Neuron*, *59*(4), 530–545.

Hudspeth, A., Jülicher, F., & Martin, P. (2010b). A critique of the critical cochlea: hopf—a bifurcation—is better than none. *Journal of neurophysiology*, *104*(3), 1219–1229.

Islam, M. A., Xu, Y., Monk, T., Afshar, S., & van Schaik, A. (2022). Noise-robust text-dependent speaker identification using cochlear models. *The Journal of the Acoustical Society of America (JASA)*, *151*(1), 500–516. https://doi.org/10.1121/10.0009314

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural computation*, *3*(1), 79–87.

Johannesma, P. (1972). The pre-response stimulus ensemble of neurons in the cochlear nucleus. *Symposium on Hearing Theory, 1972*.

Jordan, M. I., & Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural computation*, *6*(2), 181–214.

Juang, B.-H., & Rabiner, L. R. (2005). Automatic speech recognition–a brief history of the technology development. *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara*, *1*, 67.

Karuppuswamy, R., & Arumugam, K. (2013). Folded architecture for digital gammatone filter used in speech processor of cochlear implant. *ETRI Journal*, *35*(4), 697–705.

Kemp, D. T. (1978). Stimulated acoustic emissions from within the human auditory system. *The Journal of the Acoustical Society of America*, *64*(5), 1386–1391.

Kemp, D. T. (2002). Otoacoustic emissions, their origin in cochlear function, and use. *British medical bulletin*, *63*(1), 223–241. https://doi.org/10.1093/bmb/63.1.223

Kim, J. C., & Large, E. W. (2015). Signal processing in periodically forced gradient frequency neural networks. *Frontiers in computational neuroscience*, *9*, 152.

Kim, S., Hori, T., & Watanabe, S. (2017). Joint ctc-attention based end-to-end speech recognition using multi-task learning. *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 4835–4839.

Kingma, D. P., & Ba, J. (2014). Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kinouchi, O., & Copelli, M. (2006). Optimal dynamical range of excitable networks at criticality. *Nature physics*, *2*(5), 348–351.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*.

Lample, G., Conneau, A., Denoyer, L., & Ranzato, M. (2017). Unsupervised machine translation using monolingual corpora only. https://doi.org/10.48550/ARXIV.1711.00043

Large, E. W., Almonte, F. V., & Velasco, M. J. (2010). A canonical model for gradient frequency neural networks. *Physica D: Nonlinear Phenomena*, *239*(12), 905–911. https://doi.org/10.1016/j.physd.2009.11.015

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

Leonard, R. G., & Doddington, G. R. (1993). Tidigits ldc93s10 [Available from the Linguistic Data Consortium].

Li, X. L., & Liang, P. (2021). Prefix-tuning: optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Li, X., Shougat, M. R. E. U., Kennedy, S., Fendley, C., Dean, R. N., Beal, A. N., & Perkins, E. (2021). A four-state adaptive hopf oscillator. *PLOS ONE*, *16*(3), 1–14. https://doi.org/10.1371/journal.pone.0249131

Liberman, M. C., Liberman, L. D., & Maison, S. F. (2014). Efferent feedback slows cochlear aging. *Journal of Neuroscience*, *34*(13), 4599–4607.

Liu, S.-C., Van Schaik, A., Minch, B. A., & Delbruck, T. (2013). Asynchronous binaural spatial audition sensor with 2 × 64 × 4 channel output. *IEEE transactions on biomedical circuits and systems*, *8*(4), 453–464.

Lopes, C., & Perdigao, F. (2011). Phone recognition on the timit database. *Speech Technologies/Book*, *1*, 285–302.

López-Espejo, I., Tan, Z.-H., & Jensen, J. (2021). Exploring filterbank learning for keyword spotting. *2020 28th European Signal Processing Conference (EUSIPCO)*, 331–335. https://doi.org/10.23919/Eusipco47968.2020.9287772

Lyon, R. F. (2011a). Using a cascade of asymmetric resonators with fast-acting compression as a cochlear model for machine-hearing applications.

Lyon, R. F. (2011b). Cascades of two-pole–two-zero asymmetric resonators are good models of peripheral auditory function. *The Journal of the Acoustical Society of America*, *130*(6), 3893–3904. https://doi.org/10.1121/1.3658470

Lyon, R. F. (2017a). *Human and machine hearing: extracting meaning from sound*. Cambridge University Press. https://doi.org/10.1017/9781139051699

Lyon, R. F. (2017b). *Human and machine hearing: extracting meaning from sound*. Cambridge University Press.

Maison, S. F., Usubuchi, H., & Liberman, M. C. (2013). Efferent feedback minimizes cochlear neuropathy from moderate noise exposure. *Journal of Neuroscience*, *33*(13), 5542–5552.

Martin, P., & Hudspeth, A. (1999). Active hair-bundle movements can amplify a hair cell's response to oscillatory mechanical stimuli. *Proceedings of the National Academy of Sciences*, *96*(25), 14306–14311.

Meddis, R. (1986). Simulation of mechanical to neural transduction in the auditory receptor. *The Journal of the Acoustical Society of America*, *79*(3), 702–711.

Millet, J., Caucheteux, C., Orhan, P., Boubenec, Y., Gramfort, A., Dunbar, E., Pallier, C., & King, J.-R. (2022). Toward a realistic model of speech processing in the brain with self-supervised learning. https://doi.org/10.48550/ARXIV.2206.01685

Moore, B. C., & Glasberg, B. R. (1983). Suggested formulae for calculating auditory-filter bandwidths and excitation patterns. *The journal of the acoustical society of America*, *74*(3), 750–753.

Morgan, N., & Bourlard, H. (1990). Continuous speech recognition using multilayer perceptrons with hidden markov models. *International conference on acoustics, speech, and signal processing*, 413–416.

Munoz, M. A. (2018). Colloquium: criticality and dynamical scaling in living systems. *Reviews of Modern Physics*, *90*(3), 031001.

Neely, S. T. (1993). A model of cochlear mechanics with outer hair cell motility. *The journal of the acoustical society of America*, *94*(1), 137–146.

Ngamkham, W., Sawigun, C., Hiseni, S., & Serdijn, W. A. (2010). Analog complex gammatone filter for cochlear implant channels. *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, 969–972. https://doi.org/10.1109/ISCAS.2010.5537383

Nobili, R., Mammano, F., & Ashmore, J. (1998). How well do we understand the cochlea? *Trends in neurosciences*, *21*(4), 159–167.

Noé, P.-G., Parcollet, T., & Morchid, M. (2020). Cgcnn: complex gabor convolutional neural network on raw speech. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7724–7728.

Olive, J. P., Greenwood, A., & Coleman, J. (1993). *Acoustics of american english speech: a dynamic approach*. Springer Science & Business Media.

Oord, A. v. d., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. https://doi.org/10.48550/ARXIV.1807.03748

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., & Auli, M. (2019). Fairseq: a fast, extensible toolkit for sequence modeling. *Proceedings of NAACL-HLT: Demonstrations*. https://doi.org/10.48550/ARXIV.1904.01038

Palaz, D., Collobert, R., & Doss, M. M. (2013a). End-to-end phoneme sequence recognition using convolutional neural networks. *arXiv preprint arXiv:1312.2137*.

Palaz, D., Collobert, R., & Doss, M. M. (2013b). Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks. *arXiv preprint arXiv:1304.1018*.

Palaz, D., Doss, M. M., & Collobert, R. (2015). Convolutional neural networks-based continuous speech recognition using raw speech signal. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4295–4299.

Pan, J., Liu, C., Wang, Z., Hu, Y., & Jiang, H. (2012). Investigation of deep neural networks (dnn) for large vocabulary continuous speech recognition: why dnn surpasses gmms in acoustic modeling. *2012 8th International Symposium on Chinese Spoken Language Processing*, 301–305.

Panayotov, V., Chen, G., Povey, D., & Khudanpur, S. (2015). Librispeech: an asr corpus based on public domain audio books. *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 5206–5210.

Parcollet, T., Morchid, M., & Linares, G. (2020). E2e-sincnet: toward fully end-to-end speech recognition. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7714–7718.

Parthasarathi, S. H. K., & Strom, N. (2019). Lessons from building acoustic models with a million hours of speech. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6670–6674. https://doi.org/10.1109/ICASSP.2019.8683690

Paul, D. B., & Baker, J. (1992). The design for the wall street journal-based csr corpus. *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*.

Pedersen, P. (1965). The mel scale. *Journal of Music Theory*, *9*(2), 295–308.

Pfeiffer, J., Ruder, S., Vulić, I., & Ponti, E. M. (2023). Modular deep learning. *arXiv preprint arXiv:2302.11529*.

Probst, R., Lonsbury-Martin, B. L., & Martin, G. K. (1991). A review of otoacoustic emissions. *The Journal of the Acoustical Society of America (JASA)*, *89*(5), 2027–2067. https://doi.org/10.1121/1.400897

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286.

Ramasesh, V. V., Lewkowycz, A., & Dyer, E. (2021). Effect of scale on catastrophic forgetting in neural networks. *International Conference on Learning Representations*.

Rasmussen, G. L. (1946). The olivary peduncle and other fiber projections of the superior olivary complex. *Journal of Comparative Neurology*, *84*(2), 141–219.

Ravanelli, M., Parcollet, T., & Bengio, Y. (2019). The pytorch-kaldi speech recognition toolkit. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6465–6469. https://doi.org/10.1109/ICASSP.2019.8683713

Ravanelli, M., & Bengio, Y. (2018a). Interpretable convolutional filters with sincnet. https://doi.org/10.48550/ARXIV.1811.09725

Ravanelli, M., & Bengio, Y. (2018b). Speaker recognition from raw waveform with sincnet. *Proceedings of the IEEE Spoken Language Technology Workshop (SLT)*, 1021–1028. https://doi.org/10.1109/SLT.2018.8639585

Ravanelli, M., & Bengio, Y. (2018c). Speech and speaker recognition from raw waveform with sincnet. *arXiv preprint arXiv:1812.05920*.

Righetti, L., Buchli, J., & Ijspeert, A. J. (2005). From dynamic hebbian learning for oscillators to adaptive central pattern generators. *Proceedings of 3rd International Symposium on Adaptive Motion in Animals and Machines–AMAM 2005*.

Robinson, D. W., & Dadson, R. S. (1956). A re-determination of the equal-loudness relations for pure tones. *British Journal of Applied Physics*, *7*(5), 166.

Romero, G. E., & Trussell, L. O. (2022). Central circuitry and function of the cochlear efferent systems. *Hearing research*, *425*, 108516.

Rosenbaum, C., Cases, I., Riemer, M., & Klinger, T. (2019). Routing networks and the challenges of modular and compositional computation. *arXiv preprint arXiv:1904.12774*.

Rosenbaum, C. G. (2020). Dynamic composition of functions for modular learning.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, *1*(5), 206–215.

Russell, I., & Sellick, P. (1977). Tuning properties of cochlear hair cells. *Nature*, *267*(5614), 858–860.

Russo, M., Stella, M., Sikora, M., & Pekić, V. (2019). Robust cochlear-model-based speech recognition. *Computers*, *8*(1), 5.

Sainath, T., Weiss, R. J., Wilson, K., Senior, A. W., & Vinyals, O. (2015). Learning the speech front-end with raw waveform cldnns.

Sainath, T. N., Kingsbury, B., Mohamed, A.-r., Dahl, G. E., Saon, G., Soltau, H., Beran, T., Aravkin, A. Y., & Ramabhadran, B. (2013). Improvements to deep convolutional neural networks for LVCSR. *2013 IEEE workshop on automatic speech recognition and understanding*, 315–320.

Schneider, S., Baevski, A., Collobert, R., & Auli, M. (2019). Wav2vec: unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862*.

Seide, F., Li, G., Yu, D., et al. (2011a). Conversational speech transcription using context-dependent deep neural networks. *Interspeech*, 437–440.

Seide, F., Li, G., & Yu, D. (2011b). Conversational speech transcription using context-dependent deep neural networks. *Proceedings of the Interspeech Conference*, 437–440.

Seki, H., Yamamoto, K., Akiba, T., & Nakagawa, S. (2019). Discriminative learning of filterbank layer within deep neural network based speech recognition for speaker adaptation.

IEICE TRANSACTIONS on Information and Systems, *102*(2), 364–374. https://doi.org/10.1587/transinf.2018EDP7252

Sennrich, R., Haddow, B., & Birch, A. (2015). Neural machine translation of rare words with subword units. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1715–1725. https://doi.org/10.48550/ARXIV.1508.07909

Sevilla, J., Heim, L., Ho, A., Besiroglu, T., Hobbhahn, M., & Villalobos, P. (2022). Compute trends across three eras of machine learning. *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Shannon, B. J., & Paliwal, K. K. (2003). A comparative study of filter bank spacing for speech recognition. *Microelectronic engineering research conference*, *41*, 310–12.

Shougat, M. R. E. U., Li, X., Mollik, T., & Perkins, E. (2021). A hopf physical reservoir computer. *Scientific Reports*, *11*(1), 19465.

Shougat, M. R. E. U., Li, X., Shao, S., McGarvey, K., & Perkins, E. (2023). Hopf physical reservoir computer for reconfigurable sound recognition. *Scientific Reports*, *13*(1), 8719.

Smith, D. W., & Keil, A. (2015). The biological role of the medial olivocochlear efferents in hearing: separating evolved function from exaptation. *Frontiers in systems neuroscience*, *9*, 12.

Smith, J. O., & Abel, J. S. (1999). Bark and erb bilinear transforms. *IEEE Transactions on speech and Audio Processing*, *7*(6), 697–708.

Sporns, O., & Betzel, R. F. (2016). Modular brain networks. *Annual review of psychology*, *67*(1), 613–640.

Sridhar, D., Stakhovskaya, O., & Leake, P. A. (2006). A frequency-position function for the human cochlear spiral ganglion. *Audiology and Neurotology*, *11*(Suppl. 1), 16–20.

Sridhar D, L. P., Stakhovskaya O. (2006). A frequency-position function for the human cochlear spiral ganglion. *Audiol Neurotol*, 16–20. https://doi.org/10.1159/000095609

Steele, C. R., & Taber, L. A. (1979). Comparison of wkb and finite difference calculations for a two-dimensional cochlear model. *The Journal of the Acoustical Society of America*, *65*(4), 1001–1006.

Stevens, S. S., & Galanter, E. H. (1957). Ratio scales and category scales for a dozen perceptual continua. *Journal of experimental psychology*, *54*(6), 377.

Stevens, S. S., Volkmann, J., & Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. *The journal of the acoustical society of america*, *8*(3), 185–190.

Stoop, R., & Gomez, F. (2022). The analysis of mammalian hearing systems supports the hypothesis that criticality favors neuronal information representation but not computation. *Entropy*, *24*(4), 540.

Stoop, R., Kanders, K., Novelli, L., & Gomez, F. (2016). Novel insights into cochlear information processing. *Proceedings of the 2016 International Symposium on Nonlinear Theory and its Applications (NOLTA2016)*, 497–500.

Tabibi, S., Kegel, A., Lai, W. K., & Dillier, N. (2017). Investigating the use of a gammatone filterbank for a cochlear implant coding strategy. *Journal of Neuroscience Methods, 277*, 63–74. https://doi.org/10.1016/j.jneumeth.2016.12.004

Terreros, G., & Delano, P. (2015). Corticofugal modulation of peripheral auditory responses. *Frontiers in Systems Neuroscience, 9*. https://doi.org/10.3389/fnsys.2015.00134

Thakur, C. S., Hamilton, T. J., Tapson, J., van Schaik, A., & Lyon, R. F. (2014). Fpga implementation of the car model of the cochlea, 1853–1856. https://doi.org/10.1109/ISCAS.2014.6865519

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems, 30*.

Vieting, P., Schlüter, R., & Ney, H. (2023). Comparative analysis of the wav2vec 2.0 feature extractor. *Speech Communication; 15th ITG Conference*, 131–135.

Villalobos, P., & Ho, A. (2022). Trends in training dataset sizes [Accessed: 2024-05-31]. https://epochai.org/blog/trends-in-training-dataset-sizes

Vincent, E., Watanabe, S., Barker, J., & Marxer, R. (2016). The 4th CHiME speech separation and recognition challenge. *URL: http://spandh. dcs. shef. ac. uk/chime challenge* {*Last Accessed on 1 August, 2018*}.

Von Békésy, G. (1960). Experiments in hearing. *McGraw-Hill*.

Wang, S., Hu, Y., & Liu, S.-C. (2022). T-nga: temporal network grafting algorithm for learning to process spiking audio sensor events. *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3273–3277.

Warr, W. B., Boche, J. B., & Neely, S. T. (1997). Efferent innervation of the inner hair cell region: origins and terminations of two lateral olivocochlear systems. *Hearing research*, *108*(1-2), 89–111.

Webster, D. B. (1966). Ear structure and function in modern mammals. *American Zoologist, 6*(3), 451–466. https://doi.org/10.1093/icb/6.3.451

Xu, Y., Afshar, S., Wang, R., Cohen, G., Singh Thakur, C., Hamilton, T. J., & van Schaik, A. (2021). A biologically inspired sound localisation system using a silicon cochlea pair. *Applied Sciences, 11*(4), 1519. https://doi.org/10.3390/app11041519

Xu, Y., Thakur, C. S., Singh, R. K., Hamilton, T. J., Wang, R. M., & van Schaik, A. (2018). A fpga implementation of the car-fac cochlear model. *Frontiers in neuroscience*, *12*, 198.

Yang, M., Chien, C.-H., Delbruck, T., & Liu, S.-C. (2016). A 0.5 v 55 μw 64 × 2 channel binaural silicon cochlea for event-driven stereo-audio sensing. *IEEE Journal of Solid-State Circuits, 51*(11), 2554–2569.

Yao, Z., Wu, D., Wang, X., Zhang, B., Yu, F., Yang, C., Peng, Z., Chen, X., Xie, L., & Lei, X. (2021). Wenet: production oriented streaming and non-streaming end-to-end speech recognition toolkit. *arXiv preprint arXiv:2102.01547*.

You, Z., Feng, S., Su, D., & Yu, D. (2021). Speechmoe: scaling to large acoustic models with dynamic routing mixture of experts. *arXiv preprint arXiv:2105.03036*.

You, Z., Feng, S., Su, D., & Yu, D. (2022). Speechmoe2: mixture-of-experts model with improved routing. *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7217–7221.

Zeghidour, N., Teboul, O., Quitry, F. d. C., & Tagliasacchi, M. (2021). Leaf: a learnable frontend for audio classification. *arXiv preprint arXiv:2101.08596*.

Zeghidour, N., Usunier, N., Kokkinos, I., Schaiz, T., Synnaeve, G., & Dupoux, E. (2018). Learning filterbanks from raw speech for phone recognition. *2018 IEEE international conference on acoustics, speech and signal Processing (ICASSP)*, 5509–5513.

Zhang, B., Wu, D., Peng, Z., Song, X., Yao, Z., Lv, H., Xie, L., Yang, C., Pan, F., & Niu, J. (2022). Wenet 2.0: more productive end-to-end speech recognition toolkit. *arXiv preprint arXiv:2203.15455*.

Zhang, X., Heinz, M. G., Bruce, I. C., & Carney, L. H. (2001). A phenomenological model for the responses of auditory-nerve fibers: i. nonlinear tuning with compression and suppression. *The Journal of the Acoustical Society of America*, *109*(2), 648–670.

Zurek, P. (1981). Spontaneous narrowband acoustic signals emitted by human ears. *The Journal of the Acoustical Society of America*, *69*(2), 514–523.

Zweig, G., Lipes, R., & Pierce, J. (1976). The cochlear compromise. *The Journal of the Acoustical Society of America*, *59*(4), 975–982.

Zwicker, E. (1961). Subdivision of the audible frequency range into critical bands (frequenzgruppen). *The Journal of the Acoustical Society of America (JASA)*, *33*(2), 248. https://doi.org/10.1121/1.1908630

Zwislocki, J. (1953). Review of recent mathematical theories of cochlear dynamics. *The Journal of the Acoustical Society of America*, *25*(4), 743–751.

Louise **Coppieters**

PhD Research Assistant

Areas of specialization

Automatic speech recognition (ASR) • Signal processing • Biomedical applications

Languages

French	mother tongue
Dutch	● ● ● ● ○
English	● ● ● ● ○

Programming

python	● ● ● ●
Latex	● ● ● ●
bash	● ● ● ○
matlab	● ● ○ ○
c++	● ● ○ ○
Labview	● ○ ○ ○
java	● ○ ○ ○
modelsim	● ○ ○ ○

Current position

Oct 2020–Ongoing

Research Assistant at Idiap, PhD candidate at EPFL

Idiap research institute · Martigny, SWITZERLAND

Our research group aims to use machine learning to make inference about biological systems as well as vice-versa. To do this we develop trainable models that match current understanding of physiology. My work involves integrating our best understanding of the cochlea, a bank of Hopf oscillators, into state of the art pre-trained speech recognition models.

Experience

Jan 2024 - June 2024

Internship

Telepathy Labs · Zurich, SWITZERLAND

Education

2018–2020

Master degree in Electricity

UCLouvain · Louvain-la-Neuve, BELGIUM

Main interests: electronics, signal processing and biomedical applications. Master thesis: 'Ultra-low-power miniaturized vagus nerve sensing platform for treating refractory epilepsy'

Sep 2019–Feb 2020

Student exchange

EPFL · Lausanne, Switzerland

Faculty: Electricity

2015–2018

Bachelor degree

UCLouvain · Louvain-la-Neuve, BELGIUM

Major: Electricity, Minor: Biomedical

2009–2015

Sint-Jan Berchmanscollege

CESS · Brussels, BELGIUM

Option: Mathematics and ancient Greek

Publications

Sep 2022

Low-Level Physiological Implications of End-to-End Learning of Speech Recognition.

Interspeech · Seoul, South Corea

May 2024

Training a Filter-Based Model of the Cochlea in the Context of Pre-Trained Acoustic Models.

Acoustics ·

Volunteering

2015–Ongoing

Scouting

· Brussels, BELGIUM

2015 - 2017: Scout leader assistant

2017 - 2018: Scout leader

· Lausanne, SWITZERLAND

2019 - 2021: Scout leader

2021 - 2024: Assistant to the National Commissioner of Girl Scouts

☎ 0041 76 214 44 81 @ `louise.coppieters@gmail.com` 🌐 `https://www.idiap.ch/~lcoppieters/`