Learning representations of speech from the raw waveform

Neil Zeghidour

THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences et Lettres

PSL Research University

Préparée à l'École Normale Supérieure

Apprentissage de représentations de la parole à partir du signal brut

Learning representations of speech from the raw waveform

École doctorale n°158

ÉCOLE DOCTORALE CERVEAU-COGNITION-COMPORTEMENT (ED3C)

Spécialité APPRENTISSAGE AUTOMATIQUE

COMPOSITION DU JURY :

M Emmanuel VINCENT
Rapporteur, Président du jury
INRIA Nancy - Grand Est

Mme Tara SAINATH, Rapporteure
Google AI, New York

M Stéphane MALLAT, Membre du jury
Collège de France, ENS

M Gabriel SYNNAEVE, Membre du jury
Facebook A.I. Research, New-York

M Emmanuel Dupoux, Directeur de thèse
ENS, EHESS, INRIA, Facebook A.I. Research

M Nicolas USUNIER, Co-directeur de thèse
Facebook A.I. Research, Paris

Soutenue par **Neil Zeghidour**

le 13 Mars 2019

Dirigée par **Emmanuel DUPOUX**

et par **Nicolas USUNIER**

DEC ENS DÉPARTEMENT D'ÉTUDES COGNITIVES

PSL RESEARCH UNIVERSITY PARIS

Learning Representations of Speech from the Raw Waveform

by

Neil Zeghidour

Abstract

While deep neural networks are now used in almost every component of a speech recognition system, from acoustic to language modeling, the input to such systems are still fixed, handcrafted, spectral features such as mel-filterbanks. This contrasts with computer vision, in which a deep neural network is now trained on raw pixels. Mel-filterbanks contain valuable and documented prior knowledge from human auditory perception as well as signal processing, and are the input to state-of-the-art speech recognition systems that are now on par with human performance in certain conditions. However, mel-filterbanks, as any fixed representation, are inherently limited by the fact that they are not fine-tuned for the task at hand. We hypothesize that learning the low-level representation of speech with the rest of the model, rather than using fixed features, could push the state-of-the art even further. We first explore a weakly-supervised setting and show that a single neural network can learn to separate phonetic information and speaker identity from mel-filterbanks or the raw waveform, and that these representations are robust across languages. Moreover, learning from the raw waveform provides significantly better speaker embeddings than learning from mel-filterbanks. These encouraging results lead us to develop a learnable alternative to mel-filterbanks, that can be directly used in replacement of these features. In the second part of this thesis we introduce Time-Domain filterbanks, a lightweight neural network that takes the waveform as input, can be initialized as an approximation of mel-filterbanks, and then learned with the rest of the neural architecture. Across extensive and systematic experiments, we show that Time-Domain filterbanks consistently outperform mel-filterbanks and can be integrated into a new state-of-the-art speech recognition system, trained directly from the raw audio signal. Fixed speech features being also used for non-linguistic classification tasks for which they are even less optimal, we perform dysarthria detection from the waveform with Time-Domain filterbanks and show that it significantly improves over mel-filterbanks or low-level descriptors. Finally, we discuss how our contributions fall within a broader shift towards fully learnable audio understanding systems.

Résumé

Bien que les réseaux de neurones soient à présent utilisés dans la quasi-totalité des composants d'un système de reconnaissance de la parole, du modèle acoustique au modèle de langue, l'entrée de ces systèmes reste une représentation analytique et fixée de la parole dans le domaine temps-fréquence, telle que les mel-filterbanks. Cela se distingue de la vision par ordinateur, un domaine où les réseaux de neurones prennent en entrée les pixels bruts. Les mel-filterbanks sont le produit d'une connaissance précieuse et documentée du système auditif humain, ainsi que du traitement du signal, et sont utilisées dans les systèmes de reconnaissance de la parole les plus en pointe, systèmes qui rivalisent désormais avec les humains dans certaines conditions. Cependant, les mel-filterbanks, comme toute représentation fixée, sont fondamentalement limitées par le fait qu'elles ne soient pas affinées par apprentissage pour la tâche considérée. Nous formulons l'hypothèse qu'apprendre ces représentations de bas niveau de la parole, coinjontement avec le modèle, permettrait de faire avancer davantage l'état de l'art. Nous explorons tout d'abord des approches d'apprentissage faiblement supervisé et montrons que nous pouvons entraîner un unique réseau de neurones à séparer l'information phonétique de celle du locuteur à partir de descripteurs spectraux ou du signal brut et que ces représentations se transfèrent à travers les langues. De plus, apprendre à partir du signal brut produit des représentations du locuteur significativement meilleures que celles d'un modèle entraîné sur des mel-filterbanks. Ces résultats encourageants nous mènent par la suite à développer une alternative aux mel-filterbanks qui peut être entraînée à partir des données. Dans la seconde partie de cette thèse, nous proposons les Time-Domain filterbanks, une architecture neuronale légère prenant en entrée la forme d'onde, dont on peut initialiser les poids pour répliquer les mel-filterbanks et qui peut, par la suite, être entraînée par rétropropagation avec le reste du réseau de neurones. Au cours d'expériences systématiques et approfondies, nous montrons que les Time-Domain filterbanks surclassent systématiquement les mel-filterbanks, et peuvent être intégrées dans le premier système de reconnaissance de la parole purement convolutif et entraîné à partir du signal brut, qui constitue actuellement un nouvel état de l'art. Les descripteurs fixes étant également utilisés pour des tâches de classification non-linguistique, pour lesquelles elles sont d'autant moins optimales, nous entraînons un système de détection de dysarthrie à partir du signal brut, qui surclasse significativement un système équivalent entraîné sur des mel-filterbanks ou sur des descripteurs de bas niveau. Enfin, nous concluons cette thèse en expliquant en quoi nos contributions s'inscrivent dans une transition plus large vers des systèmes de compréhension du son qui pourront être appris de bout en bout.

Acknowledgments

I wish to thank the members of my thesis committee - Tara Sainath, Emmanuel Vincent, and Stéphane Mallat – for generously offering their time in reviewing and evaluating this manuscript and the defense to come, as well as my contributions to the field.

I thank Facebook for funding this thesis, and providing me with all the computational, human, and financial resources necessary to the realization of this work.

The rest of these acknowledgements will be in French.

Tout d'abord, je remercie Emmanuel Dupoux, mon directeur de thèse. C'est lui qui m'a initié et formé à la recherche scientifique, en m'encadrant durant mon stage de master. Il a par la suite joué un rôle déterminant dans la construction du projet de CIFRE, avant de devenir mon directeur de thèse. Sa vaste culture scientifique, sa capacité à prendre du recul bien au-delà des tendances et effets d'annonce, ainsi que son sens de la rigueur m'ont été précieux tout au long de cette thèse. Je le remercie également pour sa bienveillance et l'intérêt profond qu'il porte au devenir de ses étudiants, sa tolérance envers mon organisation approximative et mes disparitions régulières pour travailler seul pendant des périodes parfois étendues, dont il ne m'a jamais tenu rigueur.

Ensuite, je remercie Nicolas Usunier, mon co-directeur côté Facebook. J'ai rencontré Nicolas pour la première fois lors de ma première visite à Facebook durant mon stage de master. Il me fit ensuite passer l'entretien qui reste à ce jour celui qui me donna le plus de fil à retordre, avant de devenir mon directeur de thèse. Sa connaissance encyclopédique du machine learning, et son intérêt progressif pour la parole et le traitement du signal m'ont beaucoup aidé tout au long de cette thèse. C'est également devenu un ami cher, et quel que soit mon futur dans ou hors de Facebook, j'espère continuer à le retrouver régulièrement à l'Entracte.

Enfin, je remercie mon troisième encadrant, Gabriel Synnaeve, qui fut le 3ème membre de FAIR 13ème avant de s'installer outre-Atlantique. Il fut le pont entre l'ENS et FAIR dans l'établissement du projet CIFRE. Il m'introduisit par la suite à

l'équipe de reconnaissance de la parole, que je rejoins par la suite grâce à lui. Dans une équipe d'une taille somme toute modeste, j'ai beaucoup appris à ses côtés.

J'ajoute à ces remerciements à mes directeurs de thèse, peut-être avant toute autre chose, l'expression de ma profonde gratitude pour m'avoir toujours laissé libre de choisir mes problématiques de recherche, sans ne jamais chercher à m'imposer des directions qui n'étaient pas les miennes. Merci à eux de m'avoir fait confiance.

Merci à toute mon équipe au LSCP d'avoir ouvert mes horizons et éveillé ma curiosité depuis bientôt 4 ans. En particulier, merci à Adriana, Ewan, Thomas et Alexander Martin pour leur amitié. Merci aux « chad » PhDs de FAIR Paris d'avoir créé un environnement de travail aussi fun et agréable. Mention spéciale pour mes amis Guillaume « The Renegade Sweeper » Lample et Alexandre Défossez, avec qui j'ai eu le plaisir de travailler.

Je remercie également mes collègues de l'équipe FAIR Speech, en particulier Vitaliy et Ronan qui m'ont accueilli dans l'équipe et ont joué un rôle déterminant dans le succès de mes travaux.

Je remercie plus globalement tous mes collègues de FAIR Paris et Antoine notre manager, pour avoir créé et maintenu un environnement de travail bienveillant, décontracté, et à la fois si stimulant. J'exprime également ma gratitude à Pini pour sa cuisine inoubliable (le mi-cuit !) qui fut un réconfort bienvenu en temps de deadline.

Merci à ma bande, la Halq : Garbiche, Clemiche, Erbiche, Benos, Titax, Nemiche, Timix, Ralph, Pronzi, Elise, Juls, Alice, Mazout, Gauthier ; pour avoir été à mes côtés comme une deuxième famille depuis mon plus jeune âge.

Merci à Harry, mon binôme tout au long du MVA, et depuis mon acolyte dans la recherche aussi bien que dans mes aventures.

Je tiens enfin à remercier ma famille. Merci à Julie d'avoir enchanté cette thèse, même dans ses moments les plus ingrats. Merci de m'avoir patiemment fait réciter mes fiches pendant les entretiens, jusqu'à des heures très tardives, merci d'avoir pris et de toujours prendre si soin de moi. Merci de me rendre si heureux.

Je remercie mon père, Slimane, d'avoir affûté mon sens critique depuis mon plus jeune âge.

Je remercie mon petit frère Abel, d'être la personne la plus drôle et amusante que je connaisse.

Je remercie mon grand-frère Radouan, pour sa sagesse et sa bienveillance, et ses conseils toujours judicieux et mesurés.

Merci, enfin, et avant tout, à ma mère Karima, de m'avoir guidé vers la recherche, depuis mes visites enfant dans son laboratoire de l'Institut Pasteur, jusqu'à ses encouragements à quitter la finance pour reprendre des études en apprentissage automatique et devenir chercheur. Merci pour sa confiance indéfectible en ma capacité à mener cette thèse à bien. Je lui dédie ce manuscrit.

Publications

Mentioned in this manuscript

- **Zeghidour, N.**, Xu, Q., Liptchinsky, V., Usunier, N., Synnaeve, G., and Collobert, R.. (2019)
  Fully convolutional speech recognition.
  Arxiv preprint.

- Millet, J. and **Zeghidour, N.**. (2019)
  Learning the speech frontend for dysarthria detection.
  In *Proceedings of ICASSP*.

- Defossez, A., **Zeghidour, N.**, Usunier, N., Bottou, L., and Bach, F. (2018)
  SING: Symbol-to-instrument neural generator.
  In *Proceedings of NeurIPS*.

- **Zeghidour, N.**, Usunier, N., Synnaeve, G., Collobert, R., and Dupoux, E.. (2018)
  End-to-end Speech recognition from the raw waveform.
  In *Proceedings of Interspeech*.

- **Zeghidour, N.**, Usunier, N., Kokkinos, I., Schatz, T., Synnaeve, G., and Dupoux, E.. (2018)
  Learning Filterbanks from Raw Speech for Phone Recognition.
  In *Proceedings of ICASSP*.

- Lample, G., **Zeghidour, N.**, Usunier, N., Bordes, A., Denoyer, L., and Ranzato M-A. (2017)
  Fader Networks: Manipulating Images by Sliding Attributes.
  In *Proceedings of NIPS*.

- **Zeghidour, N.**, Synnaeve, G., Usunier, N., and Dupoux, E.. (2017)
  Joint Learning of Speaker and Phonetic Similarities with Siamese Networks.
  In *Proceedings of Interspeech*.

- **Zeghidour, N.**, Synnaeve, G., Maarten Versteegh, and Dupoux, E.. (2017)
  A Deep Scattering Spectrum - Deep Siamese Network Pipeline for Unsupervised

Acoustic Modeling.

In *Proceedings of ICASSP*.

Not mentioned in this manuscript

- Adi, Y., **Zeghidour, N.**, Collobert, R., Usunier, N., Liptchinsky, V., and Synnaeve, G.. (2019)

  To reverse the gradient or not: An empirical comparison of adversarial and multi-task learning in speech recognition.

  In *Proceedings of ICASSP*.

- Riad, R., Dancette, C., Karadayi, J., **Zeghidour, N.**, Schatz, T., and Dupoux, E.. (2018)

  Sampling strategies in siamese networks for unsupervised speech representation learning.

  In *Proceedings of Interspeech*.

- Rahma Chaabouni, Ewan Dunbar, **Zeghidour, N.**, and E. Dupoux.(2017)

  Learning Weakly Supervised Multimodal Phoneme Embeddings. In *Proceedings of Interspeech*.

Contents

0 Organization of the thesis **23**

I Introduction **26**

1 Learning recognition and classification systems from raw speech **27**

1.1 Towards end-to-end speech recognition 27

1.2 Prior knowledge and biases in speech features 31

1.2.1 Pre-emphasis 32

1.2.2 Windowing 33

1.2.3 The mel scale 34

1.2.4 Compressing the dynamic range 36

1.2.5 From mel-filterbanks to MFCC: the cepstrum and its derivatives 37

1.2.6 Mean-variance normalization 39

1.3 Speech recognition from the raw waveform 40

1.3.1 Related work 40

1.3.2 Contributions 43

1.4 Paralinguistic classification from the raw waveform 46

1.4.1 Related work 46

1.4.2 Contributions 49

1.5 Weakly-supervised speech modelling from the raw waveform 50

1.5.1 Related work 50

1.5.2 Contributions 54

II Weakly-supervised Learning of Speech Representations 56

2 Learning phonetic and speaker representations from pairs of words 57

2.1 Top-down learning of phonetic categories 57

2.2 Evaluating speech representations: triphone ABX tasks 59

2.3 ABnets 61

3 A deep scattering spectrum - deep siamese network pipeline for unsupervised acoustic modelling 63

3.1 Introduction 63

3.2 Baseline system 64

3.3 Scattering transform 65

3.4 Experiments 67

3.4.1 Weakly-supervised phonetic representation learning 67

3.4.2 Unsupervised phone representation learning on Buckeye and NCHLT 68

3.5 Conclusion 70

4 Disentangling speaker and phonetic information from the raw waveform 71

4.1 Introduction 71

4.2 Related work 73

4.3 Model 74

4.3.1 Input representations 74

4.3.2 Discriminability and invariance 75

4.3.3 Multi-task siamese network 76

4.3.4 Multi-task triamese network 78

4.4 Experimental setup 79

4.4.1 Evaluation setups 81

4.4.2 In-Domain Results 81

4.4.3 Out-of-domain results 84

4.5 Detailed analysis of discriminability and invariance in the networks . 86
4.6 Conclusion 87

III Learning the speech front-end for speech recognition and paralinguistic tasks 90

5 Time-Domain filterbanks: a learnable frontend of speech 93
5.1 Introduction 93
5.2 Time-Domain filterbanks 95
5.2.1 Mel-filterbanks computation 96
5.2.2 Approximating mel-filterbanks with convolutions in time . . . 96
5.3 Experiments 98
5.3.1 Setting 98
5.3.2 Different types of Time-Domain filterbanks 99
5.3.3 Results 100
5.4 Adding a learnable pre-emphasis layer 101
5.5 Analysis of learnt filters 101
5.6 Conclusion 103

6 End-to-End Speech Recognition from the Raw Waveform 105
6.1 Introduction 105
6.2 Time-Domain filterbanks and gammatone-based frontend 106
6.2.1 Low-pass filtering 108
6.2.2 Instance normalization 108
6.3 Experimental setup 109
6.3.1 Acoustic model 109
6.3.2 Variants 110
6.3.3 Hyperparameters and training 111
6.4 Experiments 111
6.4.1 Baseline results 111
6.4.2 Instance normalization 113

6.4.3 Impact of the low-pass filter . . . 113
6.4.4 Learnable frontends vs mel-filterbanks . . . 114
6.4.5 Adding a learnable pre-emphasis layer . . . 115
6.5 Conclusion . . . 115

7 Fully Convolutional Speech Recognition 117
7.1 Introduction . . . 117
7.2 Model . . . 118
7.2.1 Convolutional Frontend: Time-Domain filterbanks . . . 119
7.2.2 Convolutional Acoustic Model . . . 119
7.2.3 Convolutional Language Model . . . 120
7.2.4 Beam-search decoder . . . 120
7.3 Experiments . . . 120
7.4 Results . . . 122
7.4.1 Word Error Rate results . . . 122
7.4.2 Analysis of the convolutional language model . . . 124
7.5 Conclusion . . . 125

8 Learning to Detect Dysarthria from Raw Speech 127
8.1 Introduction . . . 127
8.2 Model . . . 130
8.2.1 Time-Domain filterbanks . . . 130
8.2.2 Per Channel Energy Normalization . . . 130
8.2.3 LSTM and Attention model . . . 131
8.3 Experimental setup . . . 131
8.4 Results . . . 133
8.4.1 Fully learnable frontend . . . 134
8.5 Conclusion . . . 135

9 Conclusion 137
9.1 Summary of the contributions . . . 137

9.2 Towards fully learnable audio understanding systems 139
9.3 Rethinking Time-Domain filterbanks: how much structure do we need? 141

A Fader Networks:Manipulating Images by Sliding Attributes 143

B SING: Symbol-to-Instrument Neural Generator 155

List of Figures

1-1 Evolution of performance on the TIMIT dataset along years, measured in Phone Error Rate (PER) and based on the performance reported in (chronological order) [Graves et al., 2013, Ming and Smith, 1998, Mohamed et al., 2009, Tóth, 2014]. The inflexion point in 2009 corresponds to the emergence of deep acoustic models. 28

1-2 Computational steps that produce mel-filterbanks and MFCC features. 31

1-3 Power spectrum of a sentence from the TIMIT dataset, before and after pre-emphasis. 32

1-4 Several window functions, including triangular, Hann, Hamming, and Blackman. 33

1-5 40 mel-filters. 35

1-6 Mel-filterbanks before and after log compression. 37

1-7 Word Error Rate (%) of Deep Speech 2 [Amodei et al., 2015] on a validation set, against the quantity of training data. 51

2-1 Visualization of a within and across-speaker ABX tasks from the minimal pair ("beg", "bag"). 59

2-2 The ABnet architecture. 61

3-1 A deep scattering spectrum with two layers. 67

3-2 From left to right: Across-speaker ABX error on TIMIT (as percentages), measured on raw features (yellow bars), best ABnet models trained on mel-filterbanks (purple bars), best ABnet models trained on scattering spectrum (blue bars), and outputs of three supervised systems (red bars) from [Synnaeve et al., 2014]. 69

4-1 A multi-ouput siamese network in the `WAV` setting. All parameters of each of the two branches at a given depth are shared. 77

4-2 A multi-ouput triamese network in the `MEL` setting. All parameters of each of the three branches at a given depth are shared. 78

4-3 Our convolutional architecture trained on the waveform. The part surrounded by the dashed rectangle is repeated 1, 2 and 3 times respectively for the Small, Medium and Big architectures. 80

4-4 Visualizations of what layers code through the network. The left barchart is a phonetic modelling network, the middle barchart is a speaker modelling network, and the right one is a multi-task network. In blue are units that code for the phonetic information, in yellow, the speaker information, in green the units that code for both, and in black the units that code for none. 86

5-1 Frequency response of filters, and output of mel-filterbanks and their time-domain approximation on a sentence of TIMIT. 95

5-2 Examples of learnt filters. Filters' real parts in blue; imaginary part in red. 102

5-3 Heat-map of the magnitude of the frequency response for initialization filters (left) and learned filters (right). 103

6-1 Training Letter Error Rate (LER) for the same acoustic model trained either on the gammatone-based frontend (left) or the Time-Domain filterbanks (right), with and without instance normalization. 109

7-1 Overview of the fully convolutional architecture. 119

7-2 Evolution of WER (%) on Librispeech with the perplexity of the language model. 124

8-1 Computational steps that produce mel-filterbanks and MFCC features. In Green are the operations that are learnable (or to which an equivalent is learnable) in Time-Domain filterbanks. Red boxes are the operations that remain fixed during training. 128

8-2 Proposed pipeline that learns jointly the feature extraction, the compression, the normalization and the classifier. 131

8-3 Detailed analysis of filters and compression function learned by the model. Left shows the new scales obtained by three independent models using TD-filterbanks, compared to mel scale. The center frequency is the frequency for which a filter is maximum. Right shows an approximation of the compression exponents obtained for the PCEN layer learned on mel-filterbanks. 134

List of Tables

1.1 A set of Low Level Descriptors (LLDs) of speech, typically used for paralinguistic classification. 48

3.1 ABX error (as percentages) on the ZeroSpeech 2015 datasets (English, Xitsonga) for the ABX within- and across-speaker tasks. The best scores for each condition are in **bold**. 68

4.1 Examples of A, B and X for both phonetic and speaker discriminability tasks. "sp_i" stands for speaker number i. 75

4.2 In-domain ABX error rates (%) in the `MEL` setting. 82

4.3 In-domain ABX error rates (%) in the `WAV` setting. 83

4.4 Number of parameters, size of time context (in miliseconds) and *speaker across phone* ABX error (%) for the best model from the `WAV` setting, the best Small model and the best model from the `MEL` setting. 84

4.5 Out-of-domain ABX error rates (%) in the `MEL` and `WAV` settings. . . . 84

5.1 Details of the layers for the Time-Domain filterbanks. 98

5.2 PER of the CNN-5L-ReLU-do0.7 model trained on mel-filterbanks and different learning setups of Time-Domain filterbanks. 99

5.3 PER (Phone Error Rate) on TIMIT, in percentages. "mel" stands for mel-filterbanks, while "TD-filterbanks" stands for Time-Domain filterbanks. All models but [Tóth, 2015] and [Van Den Oord et al., 2016] are trained in an end-to-end fashion. 100

6.1 Architectures of the two trainable filterbanks. Values of width and strides are given to match the standard settings of mel-filterbanks for waveform sampled at 16kHz. The convolution for the scattering-based architecture uses 80-real valued output channels and squared L2-pooling on the feature dimension to emulate a complex-valued convolution with 40 filters followed by a squared modulus operator. Thus, after the nonlinearity, both architectures have 40 filters. In Chapter 5, we use 1 to prevent log(0) and Hoshen et al. [2015] and Sainath et al. [2015a] use 0.01. We kept the values initially used by the authors of the respective contributions and did not try alternatives. We believe it has little impact on the final performance. 107

6.2 Results (%) on the open vocabulary task of the WSJ dataset. *(i)* SOTA – speech features: for state-of-the-art and representative baselines using speech features (mel-filterbanks, spectrograms or MFCC), *(ii)* SOTA-waveform: state-of-the-art from the raw waveform, including our own implementation of vanilla gammatone-based frontend without instance normalization, and *(iii)* our baseline and the different variants of the learnable frontends (with instance normalization) studied in this chapter. 112

6.3 Comparison of models trained with or without a learnable pre-emphasis layer. All models are initialized either with the scattering or gammatone initialization, and the pooling function is a fixed squared Hann window. 114

7.1 WER (%) on the open vocabulary task of WSJ. 122

7.2 WER (%) on Librispeech. 123

7.3 Evolution of WER (%) on Librispeech with the context size of the language model. 125

8.1 Speakers and number of recordings per set, the severity of each person is indicated after their ID: VL is Very Low, L is Low, and M is Medium. The bottom line shows the total number of Control (C) and Dysarthric (D) utterances per set. 132
8.2 UAR (%) of the attention-based model trained over different features or learnable frontends. The UAR is averaged over 3 runs and we report standard deviations. 133
8.3 UAR (%) of the attention-based model trained over different fully learnable frontends. "Only r" means that only the compression factor of PCEN is learned, while "only" α" refers to the setting in which we only learn the normalization strength. When not specified, both components are learned. The UAR is averaged over 3 runs and standard deviations are reported. 135

9.1 Preferred hyperparameters of mel-filterbanks for three audio tasks. . . 141

Chapter 0

Organization of the thesis

This thesis is organized in three parts. Part I is an introduction, while the contributions of this thesis are split between Parts II and III.

In Part I, we first present a brief summary of recent advances in speech recognition, and show how deep learning methods have progressively been integrated into every step of speech recognition systems, except for the features, that still remain fixed, handcrafted descriptors, in particular mel-filterbanks. Through a detailed description of the mel-filterbanks computation, we expose how these features include valuable prior knowledge that comes at the cost of inherent and undesirable biases. This motivates the goal of this thesis: replacing handcrafted speech features by a learnable frontend, that is trained with the rest of the model for the task at hand. We then give an overview of the previous work on speech recognition from the waveform, as well as our contributions on that topic. Beyond the purely linguistic content, a speech signal conveys a lot of information about its speaker (identity, age, emotion, etc.), called the paralanguage. Observing that paralinguistic classification systems of speech also use hardcoded features, we present the related work and our own contributions in training paralinguistic classifiers from raw speech. Finally, we describe one of the current challenges for speech technologies, low-resource languages, and present the existing work as well as our contributions in improving weakly-supervised and unsupervised speech modelling systems by enriching or learning their input representations.

The relative order of Parts II and III does not follow this order, but rather a

chronological one.

Part II adresses weakly-supervised and unsupervised learning of phonetic and speaker embeddings, from different type of features, and from the raw waveform. This was the first topic of this thesis.

Chapter 2 gives a preliminary background on evaluating embeddings of speech for a particular application (phonetic modelling, speaker modelling), as well as methods to learn such representations in a weakly-supervised or unsupervised way.

Chapter 3 investigates the impact of replacing mel-filterbanks by a richer scattering transform in a weakly-supervised phonetic modelling system.

In Chapter 4, we train a single model to separate the phonetic information and speaker characteristics from the speech signal, introducing incidentally our first models trained on the waveform.

The encouraging results on the quality of speaker embeddings trained from the waveform in Chapter 4 are however dampenened by the structural differences between mel-filterbanks and the proposed neural alternative, which leads us to develop a learnable alternative that can be compared to mel-filterbanks in controlled settings: the Time-Domain filterbanks.

Part III introduces Time-Domain filterbanks and shows results on speech recognition and paralinguistic classification. We also released a Pytorch implementation of Time-Domain filterbanks[1].

Chapter 5 describes the architecture of Time-Domain filterbanks, and presents results on a small-scale phonetic recognition task.

Chapter 6 proposes a improved version of Time-Domain filterbanks compared to the previous chapter, and also experiments with a previously proposed learnable frontend. We show on a large-vocabulary speech recognition task on the Wall Street Journal dataset, that Time-Domain filterbanks consistently outperform mel-filterbanks in equivalent conditions.

Chapter 7 introduces the first fully convolutional speech recognition system, using convolutional layers from the waveform up to the word transcription. Experiments

[1] `https://github.com/facebookresearch/tdfbanks`

on Wall Street Journal and Librispeech show state-of-the-art performance among end-to-end systems on both datasets.

In Chapter 8, we finally address paralinguistic classification. We apply our learnable frontend to a paralinguistic task, dysarthria detection, and show that our approach significantly outperforms both mel-filterbanks and low-level descriptors.

The discussion then summarizes the contributions and findings of this thesis, and addresses the questions that are still open.

The Appendix is composed of publications outside of the scope of this thesis, presented as is.

Appendix A introduces an auto-encoder of images that is trained to reconstruct images by disentangling the salient information of the image and the values of attributes directly in the latent space. As a result, after training, our model can generate different realistic versions of an input image by varying the attribute values, like changing the facial expression of a portrait, or updating the color of some objects.

Appendix B faces the challenge of neural music synthesis. We propose SING, a lightweight neural audio synthesizer for the original task of generating musical notes given desired instrument, pitch and velocity. SING can generate any note of more than 1,000 instruments, with a better quality than the previous state-of-the-art (as judged by human evaluators), and a generation speed which is 2,500 times faster.

Part I

Introduction

Chapter 1

Learning recognition and classification systems from raw speech

1.1 Towards end-to-end speech recognition

Automatic Speech Recognition (ASR), the task of transcribing a speech utterance automatically, has been historically performed using fixed, handcrafted speech features as input, the most standard pipeline being the MFCCs, for Mel-Frequency Cepstral Coefficients [Davis and Mermelstein, 1980]. Statistical speech recognition has relied on Hidden Markov Models (HMM) since its early days [Baker, 1975, Jelinek, 1976, Levinson et al., 1983, Rabiner, 1989], in particular GMM-HMM that use Gaussian Mixtures to model the speech features distribution [Juang et al., 1986]. A big shift ASR was brought with the use of deep neural networks as acoustic models.

After a GMM-HMM has been trained on a speech dataset, the Viterbi algorithm [Viterbi, 1967] is used to assign the most likely hidden state to each feature frame in the data. This hidden state can then be used as a label to train a deep neural network from the speech features. During inference, a neural language model can be combined with the acoustic model to improve the decoding [Mikolov et al., 2010]. The use of deep acoustic models critically improved the performance of automatic speech recognition systems [Hinton et al., 2012a], as seen in Figure 1-1, and these so called "DNN-HMM" (Deep Neural Network - Hidden Markov Model) systems are still

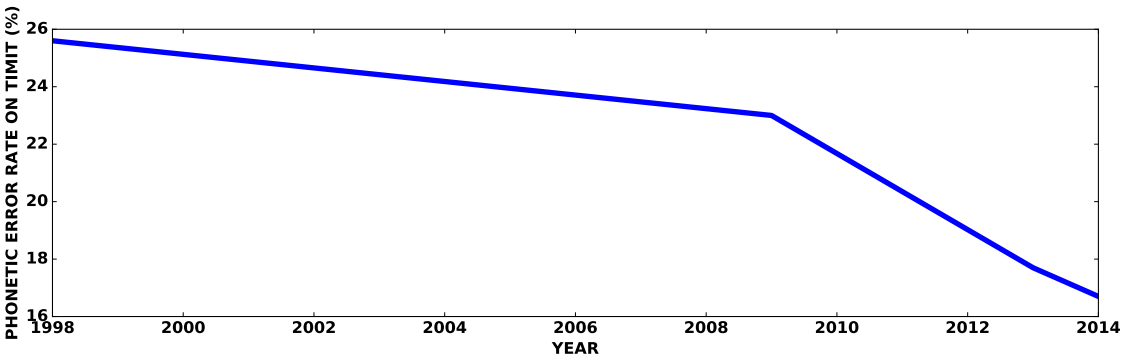


Figure 1-1: Evolution of performance on the TIMIT dataset along years, measured in Phone Error Rate (PER) and based on the performance reported in (chronological order) [Graves et al., 2013, Ming and Smith, 1998, Mohamed et al., 2009, Tóth, 2014]. The inflexion point in 2009 corresponds to the emergence of deep acoustic models.

the current state-of-the-art on almost every benchmark [Chan and Lane, 2015, Han et al., 2017, Povey et al., 2018, Tóth, 2015]. So called "connectionist" approaches had proposed using neural networks for speech recognition for decades [Bedworth et al., 1989, Bottou et al., 1989, Huckvale, 1990, Rabiner, 1989, Renals et al., 1994, Waibel et al., 1989], however it is not until the late 00's that they toppled Gaussian Mixtures.

Despite their unmatched performance, DNN-HMM systems have the drawback of requiring complex training and evaluation schemes: predicting (context-dependent) phone states requires using a pronunciation dictionary, a GMM-HMM needs to be trained to provide a deep acoustic model with forced-aligned labels, and decoding a sentence needs constructing a complex lattice with weighted finite-state transducers [Mohri et al., 2002]. This motivated the second big shift in ASR, with the emergence of end-to-end training. In this context, a deep neural network can be directly trained to predict the sequence of phonemes [Graves et al., 2013], graphemes [Amodei et al., 2015], or word pieces [Rao et al., 2017] from speech features. This removes the need for a forced-alignment obtained from training an GMM-HMM beforehand. Moreover, in the case of grapheme prediction, it also removes the need for a pronunciation dictionary, and with the addition of a special "space" character, the system can even output words directly. The first end-to-end systems relied on a new loss function, CTC for Connectionist Temporal Classification [Graves et al., 2006], which does not

require an alignment (even though a sentence-level alignment is usually used). This has allowed training ASR systems in one step instead of using the two-step schedule of DNN-HMM systems [Graves et al., 2013], and has since then been implemented in systems that are competitive with the state-of-the-art [Amodei et al., 2015, Hannun et al., 2014, Miao et al., 2015], as well as into production [Battenberg et al., 2017]. A drawback of CTC is its conditional independence assumption: conditioned on the output of the acoustic model, the probability of a character is independent from the previous predictions. This makes training and inference tractable, but this assumption may be too strong. Alternatives to CTC have been proposed to deal with this problem, like the AutoSeg loss [Collobert et al., 2016] which adds a transition matrix between output characters and can be seen as a Markovian CTC, or GramCTC which learns the set of basic units during the optimization process [Liu et al., 2017b].

A second trend in end-to-end ASR then solved this problem altogether by replacing neural networks trained with CTC by sequence-to-sequence [Sutskever et al., 2014] models, which are Recurrent Neural Networks (RNNs) that take as input the sequence of feature frames and generate the output characters one by one, the conditional dependence being modeled by the internal state of the decoder [Chan et al., 2015, Chorowski et al., 2015]. In that case, the loss function is simply a cross-entropy between the prediction and the corresponding ground truth character, and inference is performed via a beam search. This is currently the biggest trend in end-to-end speech recognition, and the gap in performance with CTC or DNN-HMMs keeps reducing [Chiu et al., 2018].

Hence, speech recognition has progressively shifted towards more end-to-end systems, concurrently with other fields of application such as natural language processing [Collobert et al., 2011] or computer vision [Krizhevsky et al., 2012]. However, one can notice that along the recent history of ASR, from GMM-HMMs to sequence-to-sequence systems, one component has remain almost unchanged: the use of hardcoded input representations of the speech signal, instead of the speech signal itself. Indeed, all the ASR systems mentioned so far have been trained on hardcoded features such as mel-filterbanks, MFCC, or spectrograms. Thus, the so-called "end-to-end" ASR

systems still need a separate feature extraction, which is not integrated into the model.

Deep neural networks changed the landscape of computer vision by allowing to train an image classifier from raw pixels [Krizhevsky et al., 2012, LeCun et al., 1998], instead of training it on hardcoded features [Bay et al., 2008, Lowe, 1999, Perronnin et al., 2010]. A convolutional neural network trained directly on the raw pixels would jointly learn all levels of representation (from low-level signal processing to high-level modeling of shapes and structures) using backpropagation, exhibiting similarities with the hierarchical processing of images that is performed in the visual cortex [Fukushima and Miyake, 1982, Hubel and Wiesel, 1962, Zeiler and Fergus, 2014].

Even though convolutional neural networks have a long history [LeCun et al., 1989, Waibel et al., 1989], their mainstream use was triggered by the unprecedented performance of AlexNet [Krizhevsky et al., 2012] in the 2012 ImageNet challenge, a deep convolutional network that reported a top-5 test error of 15.3%, outperforming by more than 10% absolute the second entry, which used hardcoded features and shallow classifiers. This performance was allowed, mainly, by three factors:

- Exploiting GPUs for an efficient training of deep and otherwise prohibitive architectures [Krizhevsky et al., 2012],

- Advances in deep learning techniques, including the use of ReLUs and dropout [Hinton et al., 2012b, Nair and Hinton, 2010],

- Access to large databases, such as ImageNet [Deng et al., 2009].

Since then, every recognition task in computer vision (classification, detection, segmentation) has switched to convolutional neural networks trained directly from the raw pixels [Farabet et al., 2013, Girshick et al., 2014]. An important observation is that the three key factors mentioned above are also valid for speech recognition: large, public training datasets are available [Panayotov et al., 2015], and both GPU computing and deep neural network architectures have been exploited for speech recognition for at least as long as they have been for computer vision [Hinton et al., 2012a, Waibel et al., 1989].

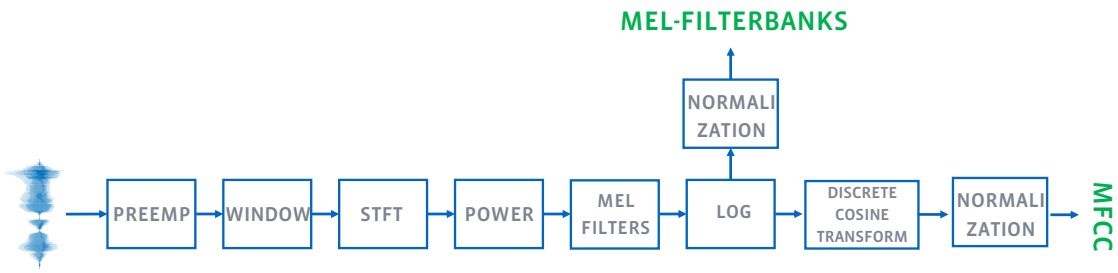
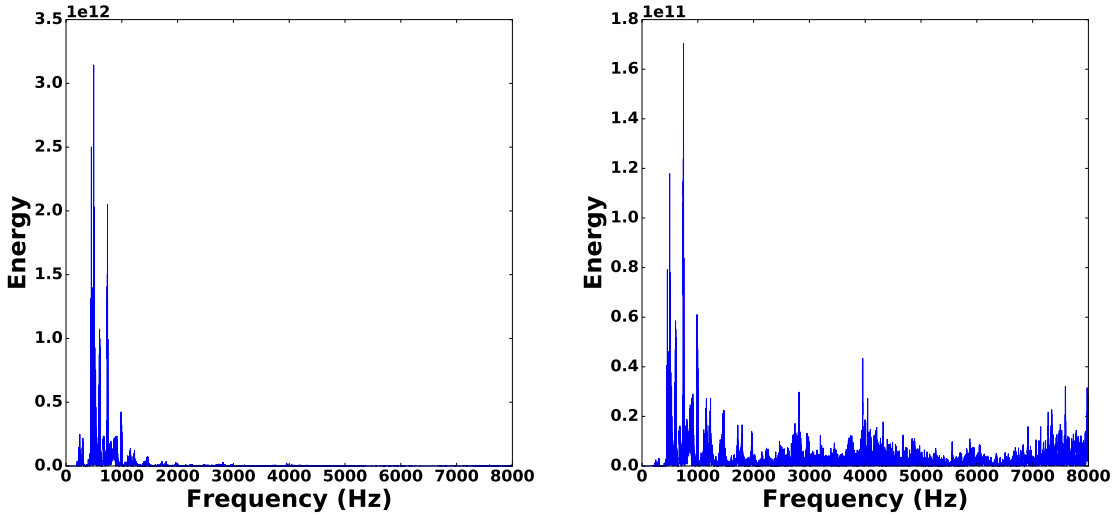


Figure 1-2: Computational steps that produce mel-filterbanks and MFCC features.

This observation motivates the main goal of this thesis: training neural networks from raw speech for recognition and classification. In the next section we describe the computation of mel-filterbanks and MFCC step by step, and show how each operation in this pipeline is inspired from prior knowledge of the auditory perception, but also how this prior knowledge can create a bias that could be corrected by making these operations learnable for the task. Then we present the existing literature on speech recognition from the waveform, as well as our contributions on that topic. Then we show how non-linguistic classification from speech is also typically performed from handcrafted features. We present the current literature on non-linguistic classification from the waveform, and our contributions to this question. Finally, we describe a current challenge for speech recognition: weakly-supervised learning for low-resource languages. We describe how weakly-supervised learning has become a necessity for the speech community, and how such weakly-supervised models can also benefit from better and learnt frontends.

1.2 Prior knowledge and biases in speech features

Speech recognition in humans shows an exceptional robustness to noise, changes in speaking style, loudness or speech rate. For this reason, the design of speech features is based on the premise that taking inspiration from the human auditory system will allow developing performing and robust ASR [Morgan et al., 2004]. This is why speech features have been designed to replicate low-level processing of speech that happens in the human inner ear. In this section, we detail the different steps that compose the



(a) Before pre-emphasis

(b) After pre-emphasis

Figure 1-3: Power spectrum of a sentence from the TIMIT dataset, before and after pre-emphasis.

computation of mel-filterbanks and MFCCs, summarized in Figure 1-2, and we show how almost every of these steps is inspired from prior knowledge about human speech perception and production, but also how this prior knowledge can cause negative bias as there is no certainty that the chosen parameters are optimal.

1.2.1 Pre-emphasis

The speech signal often carries more power in low frequencies than in higher frequencies. Moreover, speech can be contaminated by low frequency noise like DC offset or microphone pops, of which the relative energy compared to the rest of the spectrum could be attenuated. Pre-emphasis is a convolution with a first-order high-pass filter, applied to the waveform $x[n]$ to balance the energy along the spectrum:

$$y[n] = x[n] - \alpha * x[n-1], \quad 0.9 \leq \alpha < 1. \tag{1.1}$$

Figure 1-3 shows the power spectrum of a sentence of TIMIT, before and after pre-emphasis. One can notice that the distribution of energy is spread over the frequency

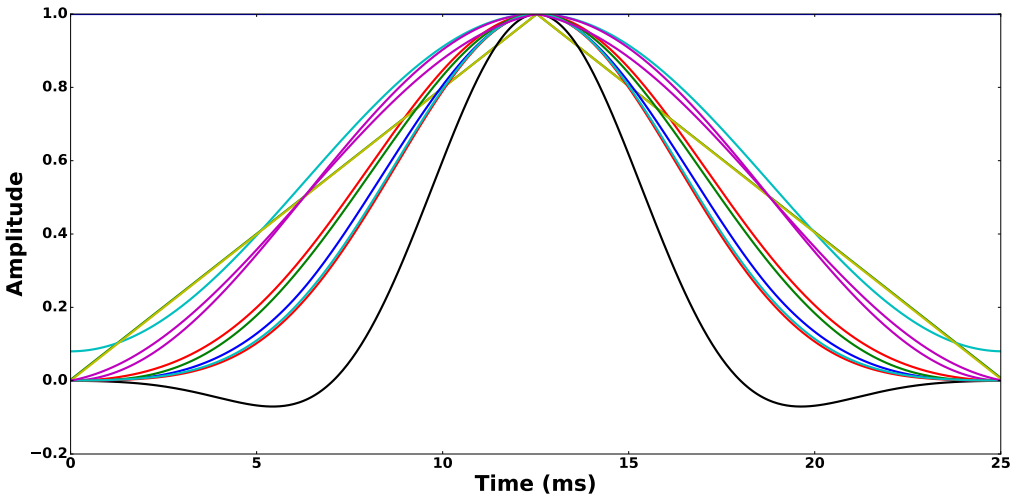


Figure 1-4: Several window functions, including triangular, Hann, Hamming, and Blackman.

axis by the pre-emphasis operation. When using spectral features, rebalancing energy along frequencies corresponds to a feature scaling, which can be critical when training a classifier or clustering algorithm on these coefficients. The α parameter controls how much energy is transferred to higher frequencies (the higher the alpha the more energy there will be in high frequencies) and the optimal value will differ depending on the recording conditions, the task, and so on. This thus required cross-validating the value of α, to reach the standard value of 0.97 in ASR. However we could expect a benefit from learning this parameter for the task at hand.

1.2.2 Windowing

Spectrograms are time-frequency representations of 1-dimensional signals, representing the energy along frequency bands and time steps. Spectrograms can not model non-stationary information inside the speech segment they are computed from, hence they are typically extracted from small overlapping speech windows in which we can make the rough assumption that the signal is stationary. Computing the Short-Term Fourier Transform (STFT) of a signal on a rectangular window (a simple cut of the speech signal in segments) will often lead to artifacts along the frequency axis (known

as spectral leakage) due to discontinuities at borders. To alleviate this problem, a window function is typically used. A window function is a multiplicative mask applied on each speech segment, of which the main characteristic is to have shrinking amplitude at its boundaries. Popular window functions are the Hamming window:

$$Hamming(n) = 0.54 - 0.46\cos(\frac{2\pi n}{N-1}), \tag{1.2}$$

with N the window size, and the Hann window:

$$Hann(n) = 0.5(1 - \cos(\frac{2\pi n}{N-1})). \tag{1.3}$$

One can notice that they only differ by the coefficient in the convex combination of the constant and the cosine function. The choice of the window function is likely to impact the performance, and the Librosa library [McFee et al., 2015] offers more than 20 different window functions to compute the STFT. Some of them are plotted in Figure 1-4. As for the pre-emphasis parameter, choosing among several window functions could be replaced by learning the appropriate one for the task at hand.

1.2.3 The mel scale

Given a waveform x sampled at f_s Hz, its spectrogram is a $D * T$ matrix, with D the number of frequency bins, linearly spaced between 0 and $f_s/2$ Hz. A *mel* function is used to map this linear scaled spectrogram to a new one, the mel scale, that is roughly linear below 1000Hz and logarithmic above. This scale warping can be implemented with the following function, proposed by O'shaughnessy [1987]:

$$Mel(f) = 1127\log(1 + \frac{f}{700}) \tag{1.4}$$

A bank of F mel-filters is derived by linearly spacing $F+1$ points on the mel-scale between 0 and $Mel(f_s/2)$, that are then mapped back to the original frequency scale, and will define the support of F triangular filters. Figure 1-5 shows the standard mel-filters from the HTK toolkit [Young et al., 2002]. The mel scale is probably the

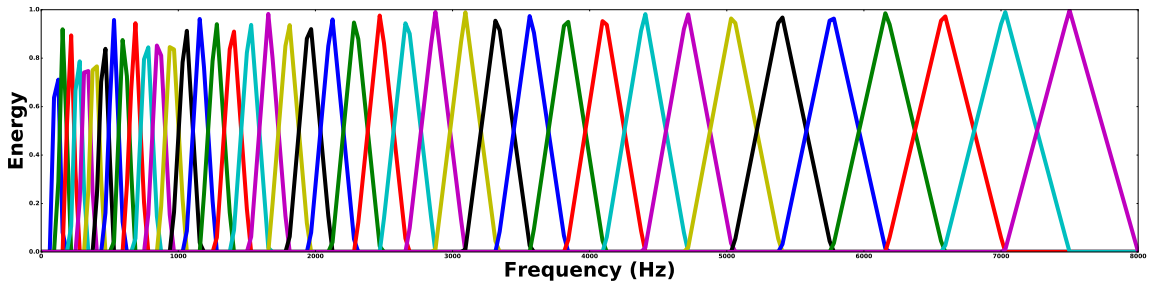


Figure 1-5: 40 mel-filters.

most popular and most standard transformation of spectrograms for speech classification or recognition, as it provides better features in most settings than a linear spectrogram. However, we can question its optimality. First, there is not only one mel scale but several, for example the implementation of Slaney [1998] that directly uses linearly spaced filters under 1000Hz and logarithmically above. Moreover, the mel scale is defined as a warping function of the frequencies, as in Equation 1.4. This function is one of many functions proposed for the mel scale since its creation [Fant, 1968, Koening, 1949, Lindsay and Norman, 2013, Makhoul and Cosell, 1976, Stevens and Volkmann, 1940]. These functions are essentially derived of a psychoacoustics experiment reported by Stevens and Volkmann [1940]. In this experiment, 10 observers are given an electric keyboard with 5 keys corresponding to 5 pure tones, and fixed lowest/highest pitch keys. Then, they have to adjust knobs controlling the tone of each key until the pitch distance between each pair of adjacent keys appears equal to their ear. This yields a few values for the mapping of the frequency scale to the mel scale, from which a continuous function such as the one in Equation 1.4 can be derived.

Umesh et al. [1999] show that many functional forms can be fitted to these discrete measurements with very good error, i.e. better than standard formulas as in Equation 1.4. Moreover, Greenwood, a student of Stevens, showed many decades later that there was a bias in the mel scale, as a different scale would be obtained from the same psychoacoustic experiments if the subjects were to listen to the tones in a descending order, rather than ascending [Greenwood, 1997]. It led him to question the validity

of the standard mel scale, which was already in 1940 proposed as a revision of a first mel scale derived a few years before by Stevens et al. [1937]. This brief historical recap of the mel scale is intended to expose its inherent flaws, as a model of human perception of pitch, but all the more as an inductive bias in machine learning systems trained to process speech:

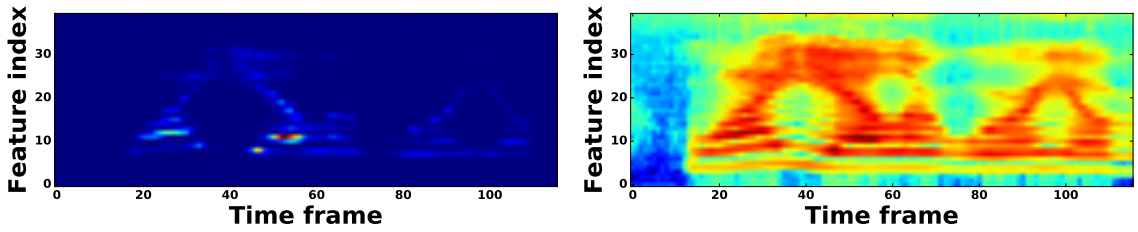
- The mel scale is a psychological scale meant to map a perceived, subjective aspect of a sound (its pitch) to a quantity that can be measured by instruments (its frequency). As a subjective scale it is based on human judgments and is inherently biased by the experimental design.

- Even if the experiments were free of any flaw, deriving more mel-filters than the number of discrete measurements made on humans requires fitting a continuous function, which will also lead to errors.

- Finally, in the context of automatic speech recognition and more widely automatic audio understanding, there is no guarantee that using the mel scale is optimal.

1.2.4 Compressing the dynamic range

A speech spectrogram can show a huge interval of variation (called the dynamic range). Human perception to loudness has been shown to be logarithmic [Fechner, 1966] or exponential with an exponent < 1 [Stevens, 1957], similarly to the way we perceive variations of pitch. This is why linearly scaled spectrograms are typically passed through a compression function to map the dynamic range to a perceptual scale. We typically use a logarithm function $x \rightarrow \log(x + \epsilon)$, with ϵ a small correction term to avoid numerical issues. Even though this function is the most standard one, other compression functions have been preferred in certain settings, such as cubic root compression [Lyons and Paliwal, 2008] or even 10th root [Schlüter et al., 2007]. This shows that, as well as for the mel-scale, there is no absolute consensus on the proper compression function to use. Moreover, again similarly to the mel-scale, even



(a) Before log compression (b) After log compression

Figure 1-6: Mel-filterbanks before and after log compression.

though these compression functions are linked to human measurements and provide good models of auditory perception, there is no guarantee that they would be optimal as part of a learning system.

1.2.5 From mel-filterbanks to MFCC: the cepstrum and its derivatives

Cascading the transformations described above yields mel-filterbanks coefficients. Computing MFCC features consists in adding a final computational step to this pipeline: a Discrete Cosine Transform (DCT), or an Inverse Short-Term Fourier Transform (ISTFT). A motivation for this operation comes from a source-filter model of speech production introduced by Fant [1970], in which speech is modeled as a convolution between a glottal excitation $e[n]$ (the source), and a vocal tract response $v[n]$ (the filter). As the response of the vocal tract characterizes the phonetic content which is produced, while the glottal pulse is linked to the pitch, we could extract good features for phonetic discriminability by "deconvolving" these two components. In the Fourier domain, this convolution becomes a product. After

the log compression, source and filter are summed as the log spectrogram computes $\omega \rightarrow \log(|E[\omega]|^2) + \log(|V[\omega]|^2)$. Considering that the formants described by V vary slowly, while the fundamental frequency and its harmonics modeled in E have faster variations, a cepstrum is obtained by computing a DCT or an ISTFT of the log spectrogram (here we do not consider the windowing and the mel filtering, for simplicity). The first coefficients correspond to the slow variations of this log spectrogram, the vocal tract, this is why the first 12 coefficients are typically extracted. They are concatenated with a log energy of the speech segment, as well as first- and second-order derivatives, to yield 39 coefficients. While both DCT and ISTFT are consistent with the source-filter model, the DCT has historically been preferred, as it yields decorrelated coefficients. The main reason is that it provided a significant advantage in terms of computational complexity when using GMM-HMM models on top of MFCC, as decorrelated features allow using diagonal covariance matrices in the GMM, reducing the number of learnable parameters from a quadratic function of the dimension of the features to a linear one.

The source-filter model of speech production described above has been validated by the extensive use of MFCC as a model of the vocal tract for speech synthesis [Airaksinen, 2012, Maia et al., 2007, Yoshimura et al., 2001, Zen et al., 2007], passed as input to a vocoder, along with an estimation of the fundamental frequency. Interestingly enough, the source-filter model was at first used to motivate the extraction of the source, i.e. the fundamental frequency or pitch [Noll, 1967, Noll and Schroeder, 1964], before focusing on the extraction of the vocal tract filter, in speech synthesis but also in speech recognition [Davis and Mermelstein, 1980]. However, using the DCT with hardcoded hyperparameters (e.g. 12 first coefficients), is also a strong inductive bias that is likely to be suboptimal for some tasks. In particular, the vocal tract fully characterizes the phonetic content in English but not in tonal languages (like Mandarin) in which the pitch modifies a phoneme's identity and its linguistic meaning. This is why learning which part of the cepstrum is relevant could improve the performance in such situations compared to using the first 12 DCT coefficients. Moreover, the source-filter model proposed above is simplistic as it does not take into

account the window function and the mel-filtering, which will impact the cepstrum in several ways (see section 6.6 of [O'shaughnessy, 1987]). First- and second order derivatives were used to incorporate information about the surrounding dynamics in a single feature vector modeled by a GMM. However, we can also question the sense of using such features as input to a deep acoustic model that can, regardless of being convolutional or recurrent, model long term dependencies (a single convolutional filter can model first-order derivatives, a second convolutional layer with one filter can model second-order derivatives).

1.2.6 Mean-variance normalization

The last step of the standard mel-filterbanks extraction pipeline is a mean-variance normalization of the coefficients, per channel. The statistics can be integrated on the entire training set, or per sequence. Even when normalizing per sequence, the statistics will change depending on whether the features are computed on a sentence's segment (e.g. when using alignments from an HMM and predicting a phonetic state) or on the full sentence (which is the case in an end-to-end setting). In offline settings, and when the recording quality is relatively constant, statistics can be aggregated on an arbitrary long period. However, when speech recognition has to be performed in an online fashion, or when the signal is contaminated by intermittent noise, it becomes necessary to aggregate statistics on short segments, using either a fixed window [Viikki and Laurila, 1998] or an exponential moving average of the statistics [Viikki et al., 1998]. There are other, more sophisticated, normalization schemes such as histogram equalization (to match the distribution of feature vectors between training and test data) [Hilger and Ney, 2006], short-time gaussianization [Xiang et al., 2002], and many others [Alam et al., 2011, Fredes et al., 2017, Kang et al., 2016, Kumar, 2015]. This diversity of normalization schemes, and the fact that they are task dependent, as for the other components of the mel-filterbanks computation, also lead to an inclination to shift towards normalizations that could be integrated into a learnable architecture. Some of the normalizations mentioned above involve some kind of training procedure, however they are trained separately from the classifier, rather than being optimized

for the final task.

In this section, we have explained how the different computational steps that compose the mel-filterbanks or MFCCs are motivated by analogies with auditory processing, or signal processing arguments, but also how they are inherently limited and could benefit from learning their parameters from the data. In the next section we describe previous work on training deep neural networks from the raw signal for speech recognition.

1.3 Speech recognition from the raw waveform

1.3.1 Related work

To the best of our knowledge, the first attempt at training a deep neural network from the raw waveform for speech recognition is [Jaitly and Hinton, 2011] in which Jaitly and Hinton train a Restricted Boltzmann machine (RBM) as a generative model of small speech segments. After training, the hidden state of this RBM is used to build features that are given as input to a phone classifier, the label being derived from a forced-alignment. This performed better than previous literature when greatly increasing the feature rate (compared to the standard rate of 10ms in speech features), but underperformed an equivalent model trained on mel-filterbanks. Note also that the feature extraction and the classification network were not trained jointly.

Then, Palaz et al. [2013b] proposed the first phone and speech recognizers trained directly from the waveform. These models were still based on an hybrid DNN-HMM system (or HMM-CRF for [Palaz et al., 2013b]), the acoustic model being trained to predict either phone classes (for phone recognition), or context-dependent phone states [Palaz et al., 2015]. When trained on the waveform, MFCCs (the baseline features) would be replaced by several blocks involving a convolutional layer, a max-pooling layer and an hyperbolic tangent. In [Palaz et al., 2013b], this approach does not reach the performance of the baseline trained on MFCCs. On the other hand when performing large vocabulary speech recognition, Palaz et al. [2015], authors find

that with a similar overall number of parameters, a CNN trained on the waveform outperforms a fully-connected network trained on MFCC. However, it is hard to relate one architecture to the other as CNNs are structurally more parameter efficient than fully connected networks. When comparing equivalent architectures on MFCC or the raw speech (feed-forward networks with rectified linear units), Tüske et al. [2014] found the model trained on the raw waveform to considerably underperform models trained on features.

In the previous works mentioned so far, the analogy between the computations made in speech features and the ones made inside their neural alternatives (RBMs, CNNs or DNNs with ReLU or hyperbolic tangent) is not clear. This motivated the concomitant work by Hoshen et al. [2015] and Sainath et al. [2015a], who introduced a learnable frontend inspired from the computation of speech features: a convolution initialized with gammatone filters is followed by a rectified linear unit a max-pooling and a log-compression. The choice of mimicking gammatone features [Schlüter et al., 2007], comes from the fact that the mel-filtering involved in the computation of mel-filterbanks and MFCCs is performed in the frequency domain, while gammatone features are computed in time domain. Hoshen et al. [2015] first use this learnable frontend in a single channel context where it underperforms mel-filterbanks. Interestingly, they show that incorporating a log compression inside the neural network significantly improves the performance, which shows the benefits of taking an auditory perspective not only to handcraft speech features, but also when designing a learnable frontend. In a second experiment, this frontend is shown to improve the Word Error Rate (WER) over mel-filterbanks in a noisy multi-channel setting, when no beamforming is performed before the computation of the handcrafted features. This shows that the implicit beamforming learned by the first convolutional layer outperforms a naive stack of the different channels, however using a delay-sum beamforming before computing mel-filterbanks yields the best performance overall.

Sainath et al. [2015a] apply the same frontend to large vocabulary speech recognition on a very large dataset (up to 40,000 hours of speech), and show that with enough data, and depending on the structure of the acoustic model, the learnable frontend

can match the performance of mel-filterbanks. As a conclusion, taking inspiration from the computation of the speech features seemed to be a promising avenue to design a neural alternative that would outperform it.

A common characteristic of the different approaches described in this section is that they integrate the feature extraction into a neural network trained to predict phone classes or (context-dependent) phone states, and thus rely on a forced-alignment provided by an GMM-HMM. Palaz et al. [2013a], besides their work on DNN-HMM described above, have also experimented with training end-to-end phone recognition systems from the raw waveform. They show that a CNN trained jointly with a CRF and taking raw speech as input can match the performance of an DNN-HMM system trained on MFCCs, however requiring 4 times more parameters in the overall model. More recently, another end-to-end approach was proposed by Tjandra et al. [2017a], based on the sequence-to-sequence framework, and taking inspiration from speech features. An encoder-decoder is trained to generate a sequence of character from the waveform. This model diverging when trained from scratch, 9 layers of convolutions (including Network-in-Network layers [Lin et al., 2013]) are pre-trained to reconstruct mel-filterbanks and then plugged below the encoder-decoder and fine-tuned jointly with it, yielding a better performance than speech features.

The conclusions of this pre-existing literature are threefold:

- Mel-filterbanks and MFCCs are very strong baselines, which explains why they are still used in state-of-the-art systems, and improving the performance by replacing these handcrafted features by neural network layers is not trivial.

- Taking inspiration from the computation of speech features to design a learnable frontend (convolutional layers, non-linearity, log compression) has led to most successes so far.

- Most of this work has been performed in the hybrid DNN-HMM setting, and the integration of a learnable frontend into an end-to-end speech recognition system is still an open question, as the necessary pre-training of Tjandra et al. [2017a] suggests that it can be hard to train.

1.3.2 Contributions

The cornerstone contribution of the Part III of this thesis is Time-Domain filterbanks: a lightweight neural network that can be used directly as an alternative to speech features, can be initialized as an approximation of mel-filterbanks, and then learnt with the rest of the architecture. We took a perspective on previously proposed learnable frontends to develop Time-Domain Filterbanks:

- Time-Domain filterbanks share the same structure (number of filters, window size, window stride) as mel-filterbanks. This allows comparing it to the speech features with an identical acoustic model and in equivalent conditions (number of parameters, training scheme). Indeed, when the neural alternative to speech features is composed of many layers [Ghahremani et al., 2016, Palaz et al., 2013b, 2015, Tjandra et al., 2017a], the comparison between the fixed baseline and the learnt frontend is hindered by a confounding factor which is the capacity of the neural network. If we want to evaluate what we can gain just from learning the low-level processing instead of keeping it fixed, we should provide a frontend that roughly involves the same number of operations as the speech features, as in [Hoshen et al., 2015, Sainath et al., 2015a].

- In order for Time-Domain filterbanks to be adopted by the community, we want to develop them such that they outperform speech features, in particular the ones that have been the most competitive when training deep acoustic models: mel-filterbanks [Hinton et al., 2012a], rather than MFCC [Ghahremani et al., 2016, Palaz et al., 2013b, 2015]. Moreover, we want the gain in performance to be consistent over acoustic model architectures and datasets. If the relative performance between the fixed features and the learnable frontend depends on the acoustic model [Sainath et al., 2015a], or the recording conditions [Hoshen et al., 2015], then the frontend is less likely to become a standard replacement.

- Since the most promising models were taking inspiration from the computation of speech features, whether it's using convolutional filtering [Palaz et al., 2013b],

gammatone initialization [Hoshen et al., 2015, Sainath et al., 2015a] or log compression [Ghahremani et al., 2016], we also choose this direction. Based on the observation mentioned in the previous section that mel-filterbanks are the best performing speech features with recent ASR systems, rather than mimicking the computation of gammatones, we design Time-Domain filterbanks such that they offer the same expressivity as mel-filterbanks and can be initialized as an approximation of these features.

- Approximating mel-filterbanks could be obtained by pre-training a convolutional network to reconstruct the mel-filterbanks as in [Tjandra et al., 2017a]. However, we show that with the proper neural network design, as well as a particular initialization, we can approximate mel-filterbanks with only two convolutional layers (rather than 9 as in [Tjandra et al., 2017a]), and without any pre-training.

In Chapter 5, we introduce the Time-Domain filterbanks. We use a time-domain approximation of mel-filterbanks proposed by Andén and Mallat [2014], the scattering transform, and implement it as a neural network. We also show how we can initialize its weights to replicate the mel-filterbanks, and then let the layers be learnt jointly with the rest of the neural architecture for the task at hand. We are particularly interested in end-to-end speech recognition, as it is the most active and promising line of work in ASR. To give a proof of concept of our approach, we perform phone recognition experiments on TIMIT, in an end-to-end setting. We show that for several architectures, models trained on Time-Domain filterbanks consistently outperform their counterparts trained on comparable mel-filterbanks. Moreover, we get our best performance by learning all frontend steps, including a learnable pre-emphasis layer. We report a Phone Error Rate (PER) which is state-of-the-art among models trained on the waveform, and competitive with the literature trained on mel-filterbanks. Finally, we study the filters obtained at convergence, to understand what characteristics they display that distinguish them from the initialization. We observe that the filters have an asymmetric impulse response, unlike the wavelets used to initialize the

weights of the first convolutional layer.

In Chapter 6, we scale from phone recognition experiments on TIMIT [Garofolo et al., 1993] (4 hours) to large vocabulary speech recognition on Wall Street Journal [Paul and Baker, 1992] (80 hours). In this chapter, we build on two alternatives for trainable replacements of mel-filterbanks that use a convolutional architecture. The first one is Time-Domain filterbanks, the second one is the frontend of Hoshen et al. [2015] and Sainath et al. [2015a], inspired from gammatone features. We propose two modifications to these architectures and systematically compare them to mel-filterbanks, on the Wall Street Journal dataset. The first modification is the addition of an instance normalization layer, which greatly improves on the gammatone-based trainable filterbanks and speeds up the training of Time-Domain filterbanks. The second one relates to the low-pass filter used in these approaches. These modifications consistently improve performances for both approaches. In particular, while the gammatone-based frontend as proposed by Hoshen et al. [2015] and Sainath et al. [2015a] significantly underperforms Time-Domain filterbanks and mel-filterbanks, modifying it with components inspired from Time-Domain filterbanks improves its performance such that the two frontends become comparable. Moreover, our modifications remove the need for a careful initialization of Time-Domain filterbanks. In particular, we show a consistent improvement in Word Error Rate (WER) of the trainable frontends relatively to comparable mel-filterbanks. This is the first time end-to-end models trained from the raw signal significantly outperform mel-filterbanks on a large vocabulary task under clean recording conditions.

In Chapter 7, we introduce the first fully convolutional speech recognition system. Current state-of-the-art speech recognition systems build on recurrent neural networks for acoustic and/or language modeling. In previous chapters (5 and 6) we combine a convolutional frontend (Time-Domain filterbanks) and a convolutional acoustic model from the raw waveform to predict letters. However, during inference, a word-level n-gram language model is used to improve the quality of the transcription. In this chapter, we present an alternative approach based solely on convolutional neural networks, leveraging recent advances in language modeling: instead of an n-gram, an

external convolutional language model is used to decode words, providing the first fully convolutional ASR system. On Wall Street Journal, our model matches the current DNN-HMM state-of-the-art, and is the current best end-to-end system. On Librispeech, we also report state-of-the-art performance among end-to-end models, including Deep Speech 2 trained with 12 times more acoustic data and significantly more linguistic data.

Overall, Part III introduces a new learnable frontend for ASR, which consistently outperforms its mel-filterbanks counterpart and is integrated in the first state-of-the-art system that does not use speech features. This leads us to extend this approach beyond speech recognition and consider other tasks. Indeed, even though our results show that mel-filterbanks are suboptimal for speech recognition, as hypothesized in Section 1.2, they have been tuned for speech recognition and therefore are biased in favor of this task. However, they are used in a wide variety of tasks that take speech as input, or more generally audio signals, tasks for which they are even less optimal than for speech recognition. These tasks are thus a promising avenue for learnable audio frontends.

1.4 Paralinguistic classification from the raw waveform

1.4.1 Related work

Paralanguage is the information conveyed by the speech signal and that does not relate to its phonetic content. A speech signal not only carries information about its linguistic content, but also about the speaker's identity, their intention, their emotional state, possible pathologies or speech impediments, as well as other physiological or anatomical characteristics (e.g. age, gender, stature). Automatic paralinguistic classification is an open scientific problem, with a very wide range of applications, and is getting more and more attention [Schuller et al., 2013a].

Automatic speaker identification has been an active field of research for decades

[Dehak et al., 2011, Kersta, 1962, Tosi et al., 1972], with applications in biometric identification and authentication, forensics, and personal assistants. Machine learning has also been used to predict other inherent or long term traits of a speaker, including their gender [Schuller et al., 2010], nativeness [Omar and Pelecanos, 2010] or likability [Weiss and Burkhardt, 2010].

There has also been interest in predicting short term characteristics of a speaker such as their emotional state [Nicholson et al., 2000], level of confidence [Pon-Barry, 2008], whether they are lying or telling the truth [Hirschberg et al., 2005], or if they are being sincere or sarcastic [Tepperman et al., 2006], which have a lot of applications, in particular for interactions with artificial agents and assistants.

Extracting information about a person from speech has also sparked the interest of the medical community. Indeed, a person's speech could be used for diagnosing or monitoring pathologies including Parkinson [Sapir et al., 2010], Huntington [Perez et al., 2018], or autism [Schuller et al., 2013b] and language development delays [Oller et al., 2010].

Like speech recognition, the field of paralinguistic classification has recently moved from composite pipelines to more end-to-end approaches, based on deep architectures [Schuller et al., 2018]. However, again similarly to speech recognition, these deep neural networks are still typically trained on handcrafted features. This includes mel-filterbanks or MFCCs described earlier, but also so called Low Level Descriptors (LLDs) [Eyben et al., 2010, Schuller et al., 2013b], that combine various signal processing extractions from the signal (e.g. energy, pitch estimation) as well as their derivatives and other various statistics [Eyben, 2015]. Table 1.1 shows a few of these features, however in recent work, by extracting many descriptors as well as several "functionals" for each of them (e.g. derivatives, mean, standard deviation, percentiles, linear regression coefficients), the number of features used to represent a small speech segment can be more than 6000 [Schuller et al., 2017]. These low level descriptors have been designed to model a wide variety of phonetic and paralinguistic patterns. Training a classifier on top of these representations will imply performing a feature selection for the task at hand, as the same sets of features are typically used for all

Feature index	Audio Descriptor	Feature index	Audio Descriptor
1	Attack	17	Spectral Slope
2	Decay	18	Spectral Decrease
3	Log-Attack Time	19	Spectral Rolloff
4	Attack Slope	20	Spectrotemporal Variation
5	Decrease Slope	21	Frame Energy
6	Temporal centroid	22	Spectral Flatness
7	Effective Duration	23	Spectral Crest
8	Frequency of Energy Modulation	24	Harmonic Energy
9	Amplitude Energy Modulation	25	Noise Energy
10	RMS-Energy Envelope	26	Noisiness
11	Autocorrelation-12 coefficients	27	Fundamental Frequency
12	Zero Crossing Rate	28	Inharmonicity
13	Spectral Centroid	29	Tristimulus
14	Spectral Spread	30	Harmonic Spectral Deviation
15	Spectral Skewness	31	Odd to Even Harmonic Ratio
16	Spectral Kurtosis	32	Hammarberg Index

Table 1.1: A set of Low Level Descriptors (LLDs) of speech, typically used for paralinguistic classification.

paralinguistic tasks, despite their fundamental differences in nature. Still, despite the variety of these descriptors, if useful information has been discarded in their computation, the classifier will not be able to retrieve it, even though it was present in the raw speech signal. Hence, low level descriptors also exhibit the flaws that we described in Section 1.2 for mel-filterbanks and MFCCs, in terms of suboptimality.

Spectral or cepstral features are also used in paralinguistic classification, however the biases that we emphasized in the case of speech recognition are even more harmful for paralinguistic classification. Indeed, taking inspiration from the auditory perception is relevant for speech recognition, as speech is produced to be intelligible for the human ear. On the other hand, for many paralinguistic tasks, including physical traits classification (age, anatomy, pathologies), there is no reason *a priori* for the human auditory system to be particularly performing. Moreover, the many hyperparameters of mel-filterbanks and MFCCs (e.g. the pre-emphasis parameter or the compression function) have been selected along the years to maximize the perfor-

mance of speech recognition, and are thus biased in favor of that task, possibly at the expense of paralinguistic classification. So, as for speech recognition, or even more, we could expect a gain in performance when learning jointly the feature extraction with the classification system.

Training paralinguistic classification systems from the raw waveform has been explored recently, in particular for emotion recognition [Sarma et al., 2018, Trigeorgis et al., 2016], speaker [Muckenhirn et al., 2018] and gender [Kabil et al., 2018]. Moreover, the 2017 edition of the popular Interspeech Computational Paralinguistics Challenge [Schuller et al., 2017] included for the first time among its baselines, besides traditional models trained on LLDs, deep neural networks trained directly on the raw waveform. This shows the growing interest of the community for this question.

A particular line of work in audio classification still exploits mel-filterbanks or spectrograms, but learns their compression and normalization components. Wang et al. [2017] introduce a new layer, the Per Channel Energy Normalization (PCEN). This module learns a compression function, as well as the range of the normalization, independently for every channel, inside a neural network. By learning it on top of mel-filterbanks (deprived of log compression or normalization) jointly with a keyword spotting model, they improve the performance of their system compared to standard mel-filterbanks. It has since then been used in noisy ASR [Battenberg et al., 2017], bird vocalization classification [Lostanlen et al., 2019] and audio event analysis [Lostanlen et al., 2019].

1.4.2 Contributions

In Chapter 8, motivated by the consistent results in ASR, we apply Time-Domain filterbanks to a paralinguistic task: dysarthria detection. We also use this task as a testbed for a new modification brought to Time-Domain filterbanks. Until this chapter, our learnable frontend replaces the pre-emphasis, spectral filtering, and windowing by learnable components, however two of its components remain fixed during training: the log compression and the mean-variance normalization. This limits the performance of our systems, as we described in Section 1.2 how the choice of the com-

pression function and the type of normalization will induce as much bias as the other components. This is why, in these final experiments, we aim to also replace these operations by learnable neural layers. We combine Time-Domain filterbanks and PCEN for dysarthria detection from sentence-level audio recordings, and propose a neural network that can learn a filterbank, a normalization factor and a compression power from the raw speech, jointly with the rest of the architecture.

Starting from a strong attention-based baseline on which mel-filterbanks outperform standard LLDs, we show that learning the filters or the normalization and compression improves over fixed features by 10% absolute accuracy. We also observe a gain over LLDs by learning jointly the feature extraction, the normalization, and the compression factor with the architecture. This constitutes a first attempt at learning jointly all these operations from raw audio for a speech classification task.

1.5 Weakly-supervised speech modelling from the raw waveform

1.5.1 Related work

As illustrated in Figure 1-1, the performance of speech recognition systems has considerably improved over the last two decades, such that it now gets close to the performance of human listeners in certain conditions [Saon et al., 2017, Xiong et al., 2016]. These conditions are mostly: English speech, spoken with a North-American accent, recorded in relatively clean conditions (no noise, telephone recording), without speech impediment. However, when these conditions are not met, the performance of current speech recognition systems significantly degrades [Amodei et al., 2015, Barker et al., 2013, 2015].

Figure 1-7, shows the evolution of the validation Word Error Rate (WER) of the Deep Speech 2 speech recognition system [Amodei et al., 2015], as the size of the training data grows. We can observe that, even though there are diminishing returns in high-resource regimes, scaling from 120 hours of training data to 12,000 reduces the

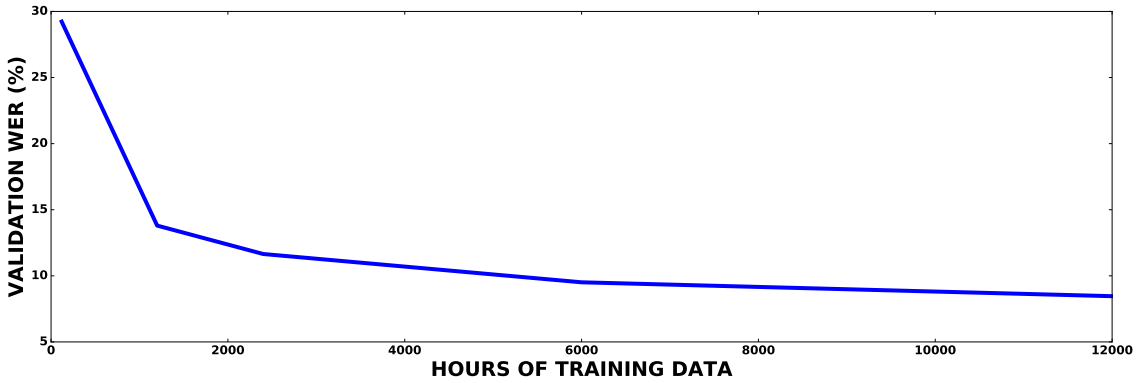


Figure 1-7: Word Error Rate (%) of Deep Speech 2 [Amodei et al., 2015] on a validation set, against the quantity of training data.

Word Error Rate from 30% to less than 9%. In that paper, Amodei et al. also report very good performance in Mandarin, for which 9,400 hours of training data are used, a language that is very different in nature from English, as it is tonal and uses an ideographic writing system. Hence, more than inherent differences between languages and the challenges they induce for acoustic modelling (tonal vs non-tonal languages) or language modelling (size of vocabulary, ideographic vs segmental writing systems), we can hypothesize that the main explanatory factor for the performance of speech technology in a language, is the availability of annotated data. Even in English, the significant difference in performance between accents (e.g. 3x higher WER in English with an Indian accent than with a North-American accent) can be explained by the dominance of North-American speakers in speech datasets.

Hence, collecting speech recordings and their transcription is the key to developing speech recognition systems. However, this process is expensive, and needs expert knowledge (from linguists or at least native speakers) to annotate recorded speech sequences. End-to-end speech recognition, as explained in Section 1.1, has allowed training speech recognition models to predict characters (graphemes), and by doing so reduced the dependency of ASR on linguistic expertise (phonetic annotations, pronunciation dictionaries). However, such models remain heavily dependent on huge amounts of character transcriptions extracted from speech recordings (e.g. DeepSpeech 2 is an end-to-end model). Moreover, when an external language model is

used to improve the quality of a speech recognition system, it usually needs to be trained on huge amounts of data. Text data is cheap to obtain in many languages, however the best language models are trained on up to billion words[1], an amount that could arduously be collected for most languages. Thus, extending a speech recognition system to a new language is expensive, and needs qualified workforce as well as vast corpora of textual data. This makes scaling to thousands of languages one of the current biggest challenges in speech technology, and may explain why the range of supported languages in assistants ranges from 3 for Amazon's Alexa to 20 in Apple's Siri [2]. These covered languages can be qualified as "high-resource languages", while we will refer to languages for which some or all resources mentioned above are lacking as "low-resource languages".

A way to face the challenge of low-resource languages is to develop algorithms that are less dependent on large, finely annotated datasets. This motivated the recent line of work on weakly-supervised and unsupervised speech recognition [Dunbar et al., 2017, Jansen et al., 2013, Versteegh et al., 2015]. These contributions generally fall in one the three following paradigms:

- Transfer learning, as a way to exploit the data available for high-resource languages to improve the performance in low-resource languages or rare accents [Toshniwal et al., 2018],

- Weakly-supervised learning, in which exhaustive and fined-grained transcriptions of sentences are replaced by coarse [Synnaeve et al., 2014], synthetic [Jia et al., 2018], or incomplete [Palaz et al., 2016] annotations,

- Unaligned supervision, in which one can leverage unrelated speech and text corpora, which is cheap as both speech and text data are easily collected independently. Inspired from recent successes in unsupervised machine translation [Artetxe et al., 2017, Conneau et al., 2017, Lample et al., 2018], Chung et al. [2018] have proposed aligning independent corpora of speech and text to train

[1] `https://catalog.ldc.upenn.edu/LDC2003T05`
[2] `https://www.globalme.net/blog/language-support-voice-assistants-compared`

word classifiers in an unsupervised setting.

An even more challenging problem is the case of languages without orthography, or unwritten dialects. Out of approximately 7,000 languages, more then 3,000 do not have a standard orthographic system [3]. This means that there is either no unique transcription for each word, or even no alphabet at all. For example, if a dataset were to be collected for Swiss German, there would be many variations in the way different annotators would transcribe the same word. This prevents training a standard speech recognition system (as the target of the learning system is ambiguous), or even computing a Word Error Rate. As speech recognition is in many cases, as in assistants, a preliminary step to natural language understanding, there has been work on bypassing it in the case of unwritten languages to map directly the speech stream to the semantics of the sentence [Dredze et al., 2010,?, Liu et al., 2017a], or a translated transcription [Bérard et al., 2016] that can then be processed by a natural language understanding algorithm in the target language.

The "zero resource" [Dunbar et al., 2017, Versteegh et al., 2015] setting refers to this extreme case in which we can not access standardized written corpora, and can only rely on untranscribed speech segments. In this case, a task that can be adressed is to discover the phonetic inventory of the language, as both written and unwritten languages are characterized by one. To that end, a particular line of work exploits an auxiliary supervision which consists in chunks of speech (typically corresponding to words) that are labeled as being identical or different [Synnaeve et al., 2014], and from which we want to discover phonetic categories. This weak supervision is interesting as it can be obtained "for free", as the output of an unsupervised algorithm. Indeed, there has been some success on training unsupervised algorithms to match chunks of speech supposedly containing the same phonetic content, whether its words, parts of words, or phrases [Park and Glass, 2008]. By combining unsupervised spoken term discovery, and weakly-supervised learning from the found segments, one can learn phonetic categories in a fully unsupervised way [Thiollière et al., 2015].

[3] `https://www.ethnologue.com/enterprise-faq/how-many-languages-world-are-unwritten-0`

1.5.2 Contributions

In the Part II of this thesis, we build on top of previous work on weakly-supervised acoustic modeling, in which we are provided with pairs of words that are either the same, or different, and use them to train a Siamese network [Bromley et al., 1993] to produce phonetically discriminative embeddings [Synnaeve et al., 2014]. We use this setting to experiment how weakly-supervised and unsupervised speech modelling can also benefit from replacing mel-filterbanks, either by a richer deep scattering spectrum, or by convolutional layers on the raw waveform.

Chapter 2 details some technical background, including the ABX evaluation of speech representations [Schatz et al., 2013], and the ABnet architecture proposed by Synnaeve et al. [2014].

In Chapter 3, we investigate the role of the input features, and in particular we test whether mel-filterbanks could be replaced by inherently richer representations derived from a deep scattering spectrum. We train a Siamese network using lexical side information similar to a well-performing architecture used in the Zero Resource Speech Challenge (2015) [Versteegh et al., 2015], and show a substantial improvement when the mel-filterbanks are replaced by scattering features. This shows that unsupervised and weakly-supervised approaches can benefit from richer features than mel-filterbanks.

These findings lead us to also experiment with training from the raw waveform in a weakly-supervised setting. Recent work [Synnaeve and Dupoux, 2014] has demonstrated, on small datasets, the feasibility of jointly learning specialized speaker and phone embeddings, using ABnets. In Chapter 4, we scale up these architectures to the 360 hours of the Librispeech corpus by implementing a sampling method to efficiently select pairs of words from the dataset and improving the loss function. We also compare the standard siamese networks fed with same (AA) or different (AB) pairs, to a 'triamese' network fed with AAB triplets. Finally, we also experiment with architectures trained directly from raw speech. We use ABX discrimination tasks to evaluate the discriminability and invariance properties of the obtained joined embed-

dings, and compare these results with mono-embeddings architectures. We find that the joined embeddings architectures succeed in effectively disentangling speaker from phoneme information, with around 10% errors for the matching tasks and embeddings (speaker task on speaker embeddings, and phone task on phone embedding) and near chance for the mismatched task. Furthermore, the results carry over to out-of-domain datasets, including a low-resource language (Xitsonga), even matching the best results obtained with similar weakly supervised techniques trained in-domain. Finally, models trained on the waveform provide significantly better speaker embeddings, suggesting that speaker identification is yet another task that can benefit from learnable frontends.

Part II

Weakly-supervised Learning of Speech Representations

Chapter 2

Learning phonetic and speaker representations from pairs of words

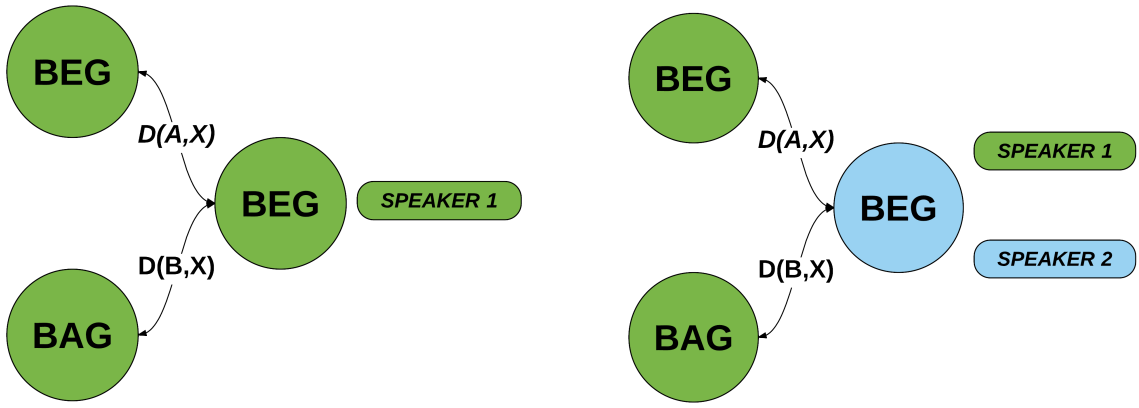
2.1 Top-down learning of phonetic categories

Unsupervised speech recognition is the task of training a system on unlabeled speech sequences, such that it can output the word transcription of a new sequence at test time. Even if no transcription is given during training, generating a word-level transcription would at least require a lexicon, which is not possible for poorly documented or unwritten languages for example. However, every language can be transcribed phonetically as long as a phonetic inventory has been collected for this language. This is why, in this part, we consider the task of learning phonetic categories in an unsupervised fashion. More precisely, given untranscribed speech utterances, we want to train a system that learns a representation of the original speech signal that separates phonetic categories, i.e. a representation space in which utterances of the same phoneme are close to each other, while utterances of different phonemes are far from each other. We call such a representation a "phonetic embedding". Projecting a speech utterance into such a space would allow an unsupervised clustering algorithm, such as K-Means [Lloyd, 1982] to find phonetic categories, such that the speech segment can be transcribed into phonemes.

Rather than training an unsupervised algorithm to directly discover phonetic classes from a speech corpus [Badino et al., 2015, Chen et al., 2015], we adopt a top-down strategy in which we exploit word-level auxiliary supervision (same or different word) to learn phonetic embeddings. The motivation for adopting this top-down approach comes from the fact that the lexicon is typically quite sparse in phonetic space. As a result, two randomly selected words will mismatch in most of their phonemes. This makes lexical clustering an easier task than phoneme clustering, and we can train unsupervised "spoken term discovery" algorithms to find repeated words or pseudo-words in unlabeled speech sequences [Kamper et al., 2015a, Park and Glass, 2008]. We can use the output of such algorithms as pseudo-labels (same/different word) to learn phonetically discriminative representations [Renshaw et al., 2015, Thiollière et al., 2015]. Thus, this top-down setting, originally weakly-supervised, can be reformulated as a fully unsupervised approach. This is consistent with the cognitive science literature, in which previous work has supported the theory of top-down phonological acquisition, that is, building an early proto-lexicon helps infants refining phoneme categories [Feldman et al., 2013, Martin et al., 2013, Swingley, 2009].

In this part we build on top of a model proposed by Synnaeve et al. [2014], where a Siamese network is trained from pairs of words labelled as "same word" or "different words", to learn phonetic embeddings from speech features. This work explores the feasability of exploiting word-level alignment in a top-down fashion, and thus uses ground truth word annotations, rather than pseudo-labels derived from a spoken term discovery algorithm. However, in Chapter 3, we also experiment with pairs produced by a spoken term discovery algorithm, in a fully unsupervised setting.

In the following sections, we describe how we evaluate the phonetic discriminability of learnt embeddings, as well as the ABnets, a particular type of Siamese network introduced by Synnaeve et al. [2014] and used throughout this part.



(a) Within-speaker ABX task.

(b) Across-speaker ABX task

Figure 2-1: Visualization of a within and across-speaker ABX tasks from the minimal pair ("beg", "bag").

2.2 Evaluating speech representations: triphone ABX tasks

A standard way of evaluating features is to train a supervised classifier and compare the classification performance to the performance we would get with a similar classifier trained on other features. However, supervised classifiers can compensate for properties of the features that would constitute considerable flaws in an unsupervised setting (e.g. poor scaling, uninformative dimensions). Hence, supervised classification performance obtained on features is not a reliable indicator of the performance of these features in an unsupervised setting, in particular for the application of clustering algorithms. We thus need to find an evaluation protocol that operates directly in the representation space, and evaluates the desired properties of a phonetic embedding. In particular, we want a representation of speech in which some distance function (e.g. Euclidean, cosine), correlates maximally with the phonetic information (utterances of the same phoneme are close to each other, and conversely), while being robust to irrelevant factors (speaker identity or channel effect should not impact the representation). These properties would allow using this representation as phonetic pseudo-labels in downstream tasks (including speech recognition). We do so by subjecting the learnt embeddings to ABX tasks [Schatz et al., 2013, Schatz, 2016].

An ABX task consists in presenting three stimuli A, B and X, with A and B belonging to different categories and X matching the category of either A or B, let us assume in this example that X belongs to the same category as A. Distances $D(A, X)$ and $D(B, X)$ are computed in the embedding space and compared. If $D(A, X) > D(B, X)$ then the score is 1 (error), else it is 0 (success). Note that D can be an actual distance (e.g. euclidean distance) or not (e.g. cosine distance, kl-divergence).

The experiments in this part are evaluated with a particular type of ABX task, adapted for speech, the triphone minimal-pair ABX task [Schatz et al., 2013]. A minimal pair is a pair of sounds, each composed of three phonemes, that only differ by their central phoneme ("beg" vs "bag"). We choose minimal pairs as they are the "worst case" kind of triphones, and being able to discriminate them would imply being able to discriminate between any triphones. A and X are two different utterances of the same triphone, while B would is an utterance of another triphone that only differs from A and X by its central phoneme. A desirable characteristic of a phonetic representation is its robustness to changes of speaker. In order to include this variability in our evaluation, we choose the utterances of these triphones such that A and X are spoken by different speakers, while B is pronounced by the same speaker as A. In this setting, observing $D(A, X) < D(B, X)$ means that the embedding space in which we compute D favors phonetic discriminability over speaker discriminability. We can also perform this task within-speaker, when A, B, and X are all produced by the same speaker. Figure 2-1 shows a visualization of the within and across speaker ABX tasks.

A global score is obtained by averaging ABX errors over all relevant triplets that can be found in the corpus on which we evaluate the representation. We obtain an error between 0% and 50% (the chance level), a low error characterizing a representation in which phonetic categories are well separated from each other. In a representation space that yields a 0% ABX error, a point is closer to any point of the same class than it is to any point of another class.

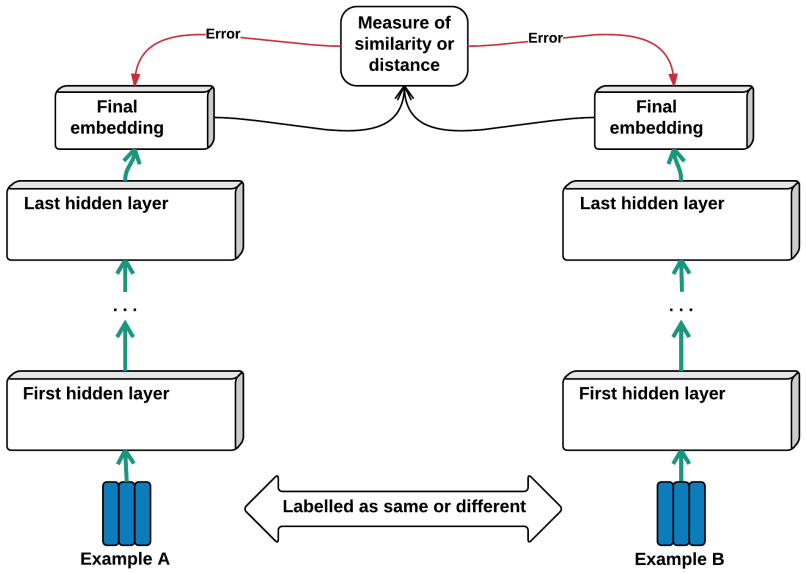


Figure 2-2: The ABnet architecture.

2.3 ABnets

Siamese networks are neural architectures that were first introduced for written signature verification [Bromley et al., 1993]. The main intuition behind these architectures is that given an abstract notion of similarity on the data we can use pairwise relations between samples to learn a representation where the distance between the embeddings of objects will reflect the abstract similarity between these objects. In other words, we want to learn a mapping $\mu(X)$ such that for a certain similarity function D in the embedding space we have $D(\mu(X_1), \mu(X_2))$ small if X_1 and X_2 are *same* objects, and large if they are *different*. Because of this architecture, the supervision required by a Siamese network consists of pairs of samples that are labeled *same* or *different*, rather than labels for individual samples.

The ABnet is composed of two copies of the same network, each copy being fed with one of the elements of a pair of input samples. These identical networks project the samples into the embedding space, through several hidden layers. A measure of similarity or distance is then computed between the two pairs depending on their relation label (*same* or *different*), and this loss function is propagated evenly in the two copies. Figure 2-2 shows a Siamese network.

An ABnet is a particular case of Siamese neural network. It uses pairs of words to learn a representation of phones. Once words are paired, the feature frames that constitute them are aligned with Dynamic Time Warping (DTW) [Sakoe and Chiba, 1978] and the paired feature frames are fed to the ABnet. As mentioned in Section 2.1, these pairs can either be extracted using ground-truth annotations, or from the output of an unsupervised spoken term discovery algorithm.

Chapter 3

A deep scattering spectrum - deep siamese network pipeline for unsupervised acoustic modelling

This chapter is based on the material from A Deep Scattering Spectrum - Deep Siamese Network Pipeline for Unsupervised Acoustic Modelling [Zeghidour et al., 2016b], accepted for oral presentation at ICASSP 2016 and a joint work with Gabriel Synnaeve, Maarten Versteegh and Emmanuel Dupoux. This work was done during my master internship at ENS under the supervision of Emmanuel Dupoux.

3.1 Introduction

Synnaeve et al. [2014] train shallow and deep ABnets on word pairs extracted from the TIMIT dataset (American English) [Garofolo et al., 1993]. The resulting models are evaluated on an ABX minimal pair discrimination task and show a phonetic discriminability which is much better than the original mel-filterbanks they are trained on, and close to supervised baselines that are trained using phonetic-level annotations.

However, regardless of the representation power of the ABnets, i.e. their depth and structure, their performance remains inherently limited by the amount of information available in their input representation. Fine-grained frequency information is lost in the computation of mel-filterbanks, mainly when averaging the spectrogram over the filters, and this loss of

information puts an upper bound on the final performance of classifiers that are trained on these features, if this fine-grained information is relevant. Training from the raw waveform would address this problem, however training a deep neural network from raw speech comes with challenges and adjustments that will be brought up in Chapter 4, and then deeply studied in Part III. Here, and as a preliminary study, we strike a middle ground by replacing the filterbanks by a deep scattering spectrum [Andén and Mallat, 2014], a representation that has many of the desirable properties of mel-filterbanks, i.e. it is stable to deformation and local translation, while retaining more information.

In this chapter, we show that both in a weakly supervised setting with gold word-level annotations on the TIMIT corpus and in a purely unsupervised setting on the Buckeye (American English) and NCHLT (Xitsonga) corpora, combining the scattering spectrum and the ABnet significantly improves the learnt representation, as evaluated with ABX errors, compared to identical ABnets trained on mel-filterbanks. Our best system even outperforms supervised GMM-HMM posteriorgrams trained with phonetic labels.

In the following sections, we first describe the baseline model used in [Synnaeve et al., 2014]. Then we explain how a deep scattering spectrum can be computed to retrieve the information that is lost in the computation of mel-filterbanks. We then present our experimental protocol and the results both in a weakly-supervised and an unsupervised setting. Finally, we summarize our findings and discuss how it leads us to training models directly from raw speech in Chapter 4.

3.2 Baseline system

The base system is the model of Synnaeve et al. [2014] a Siamese network trained on pairs of words. The pairs labeled as "same" are extracted from the ground-truth annotations. The "different" pairs are sampled randomly. Even if there is a risk of false negatives i.e. labeling frames as "different" while they actually have the same phonetic content, this probability is relatively low due to the distribution of the 39 phonemes inside the English language. 40 mel-filterbanks are computed from the original speech signal. Frames corresponding to each utterance of the pair are aligned with a Dynamic Time Warping [Sakoe and Chiba, 1978] and the resulting pairs of frames are padded with 3 context frames on each side. The resulting 7 frames are given as input to each entry of the Siamese network, respectively x

and x'. The Siamese network is given the triplet (x, x', y) where $y \in \{0, 1\}$ is 1 if x and x' are stacks of frames extracted from words labeled as "same", and 0 otherwise. Given inputs x and x', the network outputs respectively the embeddings $\mathbf{e}(x) = \mathbf{e}$ and $\mathbf{e}(x') = \mathbf{e}' \in R^d$. The similarity between embeddings is measured by their cosine $\cos(\mathbf{e}, \mathbf{e}') = \frac{\mathbf{e}.\mathbf{e}'}{||\mathbf{e}||_2||\mathbf{e}'||_2}$, and the network is trained to minimize the following loss function:

$$\ell(\mathbf{e}, \mathbf{e}', y) = \begin{cases} -\cos\left(\mathbf{e}, \mathbf{e}'\right) & \text{if } y = 1 \\ \cos(\mathbf{e}, \mathbf{e}')^2 & \text{if } y = 0 \end{cases}, \tag{3.1}$$

Hence, when two inputs are "same", we maximize their cosine similarity. However, when they are "different", we minimize the squared cosine similarity, as suggested by Synnaeve et al. [2014]. They justify this approach by the fact that minimizing the cosine similarity would draw the embeddings towards anti-colinearity, which is harder to achieve than orthogonality (which is reached when minimizing the squared cosine), and often leads to divergence. We also adopt this loss function as we want to compare the performance of different speech features, without confounding factors such as changes in the architecture, the loss function, or the optimization algorithm. However, in Chapter 4, we show that minimizing the cosine similarity down to a margin hyperparameter solves the convergence problem.

3.3 Scattering transform

For a signal x, and using the notation of Andén and Mallat [2014], we define the following wavelet transform Wx as a convolution with a low-pass filter ϕ and higher frequency complex analytic wavelets ψ_{λ_1}:

$$Wx = (x \star \phi(t), x \star \psi_{\lambda_1}(t))_{t \in R, \lambda_1 \in \Lambda_1} \tag{3.2}$$

We apply a modulus operator to the wavelets coefficients to remove the phase and extract Hilbert envelopes at different resolutions:

$$|W|x = \Big(x \star \phi(t)\,,\, |x \star \psi_{\lambda_1}(t)|\Big)_{t \in R, \lambda_1 \in \Lambda_1} \tag{3.3}$$

$S_0 x = x \star \phi(t)$ is locally invariant to translation thanks to the time averaging ϕ. This time-averaging loses the high frequency information, which is retrieved in the wavelet modulus coefficients $|x \star \psi_{\lambda_1}|$. However, these wavelet modulus coefficients are not invariant to translation, and as for S_0 a local translation invariance is obtained by a time averaging, which defines a first layer of scattering coefficients:

$$S_1 x(t, \lambda_1) = |x \star \psi_{\lambda_1}| * \phi(t) \tag{3.4}$$

Andén and Mallat [2014] show that if the wavelets ψ_{λ_1} have the same frequency resolution as the standard mel-filters, then the $S_1 x$ coefficients approximate the mel-filterbanks coefficients. They also use this characterization to explain how mel-filterbanks are provably stable to deformations of the signal (time-warping), which is interesting as they justify the design choices of mel-filterbanks with mathematical arguments, rather than psychoacoustics. Incidentally, this approximation of mel-filterbanks will be the key to designing Time-Domain filterbanks in Chapter 5, however we are here interested in deriving richer fixed features from mel-filterbanks, rather than a learnable architecture. Andén and Mallat explain that, unlike with the standard mel-filterbanks computation, we here have a strategy to recover the lost information, by passing the wavelet modulus coefficients $|x \star \psi_{\lambda_1}|$ (taken before averaging) through a bank of higher frequency wavelets ψ_{λ_2}:

$$|W_2| \, |x \star \psi_{\lambda_1}| = \Big(|x \star \psi_{\lambda_1}| \star \phi \, , \, ||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}| \Big)_{\lambda_2 \in \Lambda_2} \tag{3.5}$$

This second layer of wavelet modulus coefficients is still not invariant to translation, hence we average these coefficients with a low-pass filter ϕ to derive a second layer of scattering coefficients:

$$S_2 x(t, \lambda_1, \lambda_2) = ||x \star \psi_{\lambda_1}| \star \psi_{\lambda_2}| \star \phi(t) \tag{3.6}$$

Repeating these successive steps of computing invariant features and retrieving lost information leads to the scattering spectrum, as seen in Fig. 3-1, however speech signals are almost entirely characterized by the first two layers of the spectrum, that is why a two layers spectrum is typically used for speech representation. By concatenating the coefficients of each layer S_0, S_1 and S_2 we obtain a representation which is stable to deformations, while keeping more fine-grained information than the mel-filterbanks coefficients.

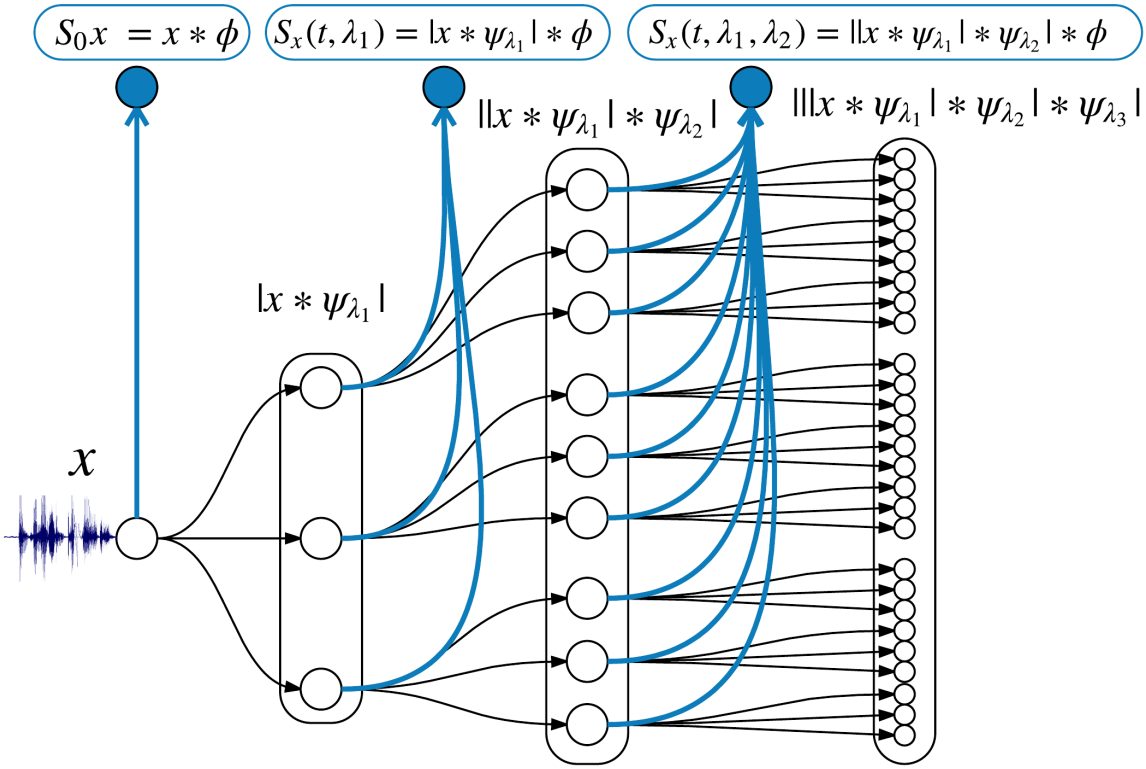


Figure 3-1: A deep scattering spectrum with two layers.

3.4 Experiments

In the following experiments, we replace mel-filterbanks by a two-layer scattering spectrum computed with a 16 ms lowpass filter, with normalized second order coefficients and log-frequency scattering [Andén and Mallat, 2014]. The features are computed with the ScatNet toolbox [Andén and Mallat, 2014]. We train our models in the same conditions as Synnaeve et al. [2014], using Adadelta [Zeiler, 2012] and early stopping. We use word-level transcriptions for training, and phone-level transcriptions for ABX evaluations.

3.4.1 Weakly-supervised phonetic representation learning

The TIMIT dataset [Garofolo et al., 1993] is a corpus of clean read speech containing a set of 10 sentences read by 630 speakers of eight major dialects of American English. All the words of more than 5 characters that are repeated in the corpus are extracted and matched as pairs of "same". This yields 62,625 pairs of same words represented as time bounding-boxes in the signal. We extract the scattering features within these boxes and align them with Dynamic

Model	English		Xitsonga	
	within	across	within	across
Baseline (MFCC)	15.6	28.1	19.1	33.8
Topline (Supervised)	12.1	16.0	3.5	4.5
FbanksABnet [Thiollière et al., 2015]	12.0	17.9	11.7	16.6
Deep ScatABnet	11.3	17.1	12.5	16.2
Shallow ScatABnet	11.0	17.0	12.0	**15.8**
DPGMM [Chen et al., 2015]	**10.8**	**16.3**	**9.6**	17.2

Table 3.1: ABX error (as percentages) on the ZeroSpeech 2015 datasets (English, Xitsonga) for the ABX within- and across-speaker tasks. The best scores for each condition are in **bold**.

Time Warping (DTW), yielding 6.77M feature frames. The pairs of "different" objects are not aligned with DTW but just aligned on the shortest one. For evaluation, we use the 39 phoneme set of Lee and Hon [1988].

Fig. 3-2 shows ABX errors on the across-speaker task. The distances used for the ABX tasks are the cosine distance for the raw features, and the symmetric KL-divergence for all trained models. "Shallow" models have one hidden layer while "Deep" models have three. All hidden layers have 200 hidden units, and the final embedding has a dimension of 100. Even though raw scattering features do not yield a better ABX error than mel-filterbanks, their use as an input representation leads to a substantial improvement after training an ABnet, with a best error of 9.8% against 11.8% for the best ABnet trained on mel-filterbanks. Our best Scattering-ABnet model even gives a better ABX score than the HMM-GMM posteriorgrams (11%), very close to the output of the deep supervised network (9.6%). In fact, changing the input representation of the Shallow ABnet from mel-filterbanks to scattering coefficients has an impact on the ABX error (from 12.4% to 10.2%) that is 3.7 times higher than adding hidden layers to get a Deep ABnet (from 12.4% to 11.8%).

3.4.2 Unsupervised phone representation learning on Buckeye and NCHLT

In this experiment we run our model under the conditions of the Zero Resource Speech Challenge 2015 [Versteegh et al., 2015]. One of the tasks in this challenge was unsupervised acoustic modeling. The challenge provided two datasets (a subset of the Buckeye Corpus

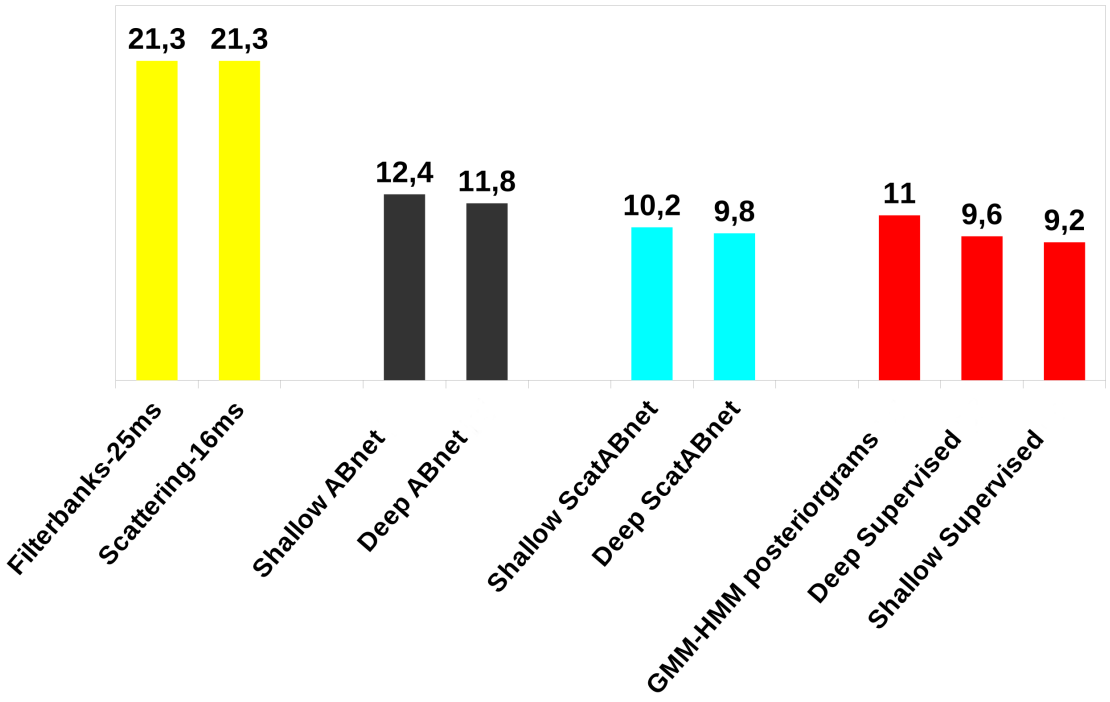


Figure 3-2: From left to right: Across-speaker ABX error on TIMIT (as percentages), measured on raw features (yellow bars), best ABnet models trained on mel-filterbanks (purple bars), best ABnet models trained on scattering spectrum (blue bars), and outputs of three supervised systems (red bars) from [Synnaeve et al., 2014].

of conversational English [Pitt et al., 2007] and a subset of the NCHLT corpus of Xitsonga [Vries et al., 2014]) as well as baseline and topline ABX scores for both datasets, both for within- and across-speakers. The baselines provided by the challenge are MFCCs, the toplines are supervised HMM-HMM posteriorgrams. In the spirit of the challenge, we extract the pairs of speech segments used for training our model in an unsupervised manner. That is, rather than taking matching words from the gold transcription, we extract them from the signal by an unsupervised spoken term discovery (STD) algorithm [Jansen and Van Durme, 2011]. This algorithm discovered 3149 pairs of similar segments of speech from the English corpus and 1782 pairs from the Xitsonga corpus, 50% being used for training and 50% for early stopping. These pairs form the "same" input to the ABnet. Like in our weakly-supervised experiments, the "different" input is composed of randomly matched segments of speech. Here, the "Deep" ScatABnet architecture consists of 2 layers of 500 nodes, with a sigmoid activation function, exactly as in a previously published study using an ABnet with filterbanks [Thiollière et al., 2015]. The "Shallow" one has only one hidden layer.

In Table 3.1, we compare our model against the challenge baselines (MFCC) and toplines (supervised HMM-GMM posteriorgrams) and also against the best performing system submitted to the challenge [Chen et al., 2015], a DPGMM system that takes as input speaker-normalized MFCCs. We observe that both ABnet variants perform better than the baseline. For English within-speaker, both systems actually outperform the supervised topline. ScatABnet has lower error scores than the FbanksABnet on all conditions except Xitsonga within-speaker. The table further shows that ScatABnet is competitive with the state of the art system of [Chen et al., 2015], in one case, across-speaker for Xitsonga, producing the lowest ABX error. These performances are remarkable given that the number of pairs on which the ABnet is trained is much lower than for the TIMIT. This low number of pairs can also explain why here a shallow architecture with fewer parameters gives a higher performance than a deep one.

3.5 Conclusion

This chapter shows that feeding rich input representations to a weakly-supervised acoustic model offers a significant leverage in terms of phonetic discriminability. The experiments on TIMIT in section 3.4.1 show that switching from standard mel-filterbanks to the scattering spectrum yields a substantial gain (about 17% in relative error rate), a higher gain than switching from a shallow to a deep network. This shows that in acoustic representation learning, putting more emphasis on the input representation might give a larger performance increase than improving the learning architecture. These results suggest that deep architectures that are trained on standard mel-filterbanks may not be exploited to their full potential, as previously shown in a supervised setting by Peddinti et al. [2014]. However, there is no guarantee that the alternative that we use, the deep scattering spectrum, is itself optimal. These findings, associated to the arguments developed in Section 1.2 on the suboptimality of fixed features, support the idea of removing speech features from acoustic models: if we observe that we can significantly outperform mel-filterbanks, then we should replace them, and rather by a learnable architecture than by other speech features. This motivates the next chapter, in which we train weakly-supervised systems directly from the waveform.

Chapter 4

Disentangling speaker and phonetic information from the raw waveform

This chapter is partly based on the material from Joint Learning and Speaker and Phonetic Similarities with Siamese Networks [Zeghidour et al., 2016a], accepted for oral presentation at Interspeech 2016 and a joint work with Gabriel Synnaeve, Nicolas Usunier and Emmanuel Dupoux. Experiments with models trained on the waveform were performed at the same period and are published for the first time in this manuscript.

4.1 Introduction

As described in Section 1.4, the speech signal carries a lot of information beyond the linguistic content, and in this Chapter we make the rough assumption that a speech signal is essentially the "entanglement" of the linguistic content and the intrinsic speaker characteristics (we do not consider other sources of variations such as channel effect, or prosody). Automatic Speech Recognition (ASR) consists in extracting linguistic information from the speech features, independently of the identity of the speaker, conditions of recording, and other irrelevant information. On the other hand, speaker identification requires extracting information from the signal that characterizes the speaker regardless of the content of their production. Hence, if we see the speech signal as the combination of orthogonal informations (linguistic content, speaker identity, noise, etc.) present in the input features, we can observe that performing any speech classification task relies on extracting one of these infor-

mations and removing the others. A limitation of this approach is that it requires individual training pipeline for each task, each pipeline extracting its relevant information from the same speech features. However, we can overcome this limitation with multi-task learning [Caruana, 1998]. Indeed, since the speech features are shared across all tasks, instead of learning separately to perform each task at the expense of all the others, we propose to learn multiple tasks at the same time from these shared speech features, a process that can be described as "disentangling" speaker identity and phonetic content.

In a weakly-supervised setting, we want to learn both phonetic and speaker representations from pairs or triplets of word-level utterances, labelled as being the same word or different ones, and being spoken by the same person or different ones. Taking inspiration from the approach detailed in Chapter 2, we use Siamese networks that we train with either a contrastive loss on pairs of words, or a ranking loss on triplets of words. A natural choice for input features to such models is mel-filterbanks since, as shown in Section 1.4, they are used in a wide range of linguistic or paralinguistic tasks (e.g. speech recognition, speaker recognition, emotion recognition). However, we are also interested in learning phonetic and speaker representations from the raw speech signal. In Section 1.2, we showed that even though mel-filterbanks are linked to human perception and should efficiently code information for a wide range of tasks (speech recognition, speaker recognition), the tuning of their many hyperparameters have most likely been tuned for speech recognition. If such a bias exists in speech features, that would favor speech recognition at the expense of performance for other tasks, we can expect speaker recognition to benefit from learning its own features from the raw waveform. This motivates the use of convolutional networks on the speech signal, which consists in replacing the speech features computation usually performed in the frequency-domain by real-valued convolutions in the time-domain.

We show that we can effectively train a single model to disentangle phonetic and speaker informations on an large scale English dataset, Librispeech [Panayotov et al., 2015], either from mel-filterbanks or from the waveform. We show that this disentanglement can generalize out-of-domain, to another English dataset (TIMIT) and even to a low-resource language, Xitsonga, spoken mostly in South Africa and Mozambique. Finally we show that both joint learning and training from the raw waveform benefit the speaker modelling task, and we perform an analysis of our networks to understand how the joint modelling is learned within the hidden layers.

In the remainder of this chapter, we first discuss in more details the related work on multi-task learning and triplet losses in Section 4.2. We then present the models in Section 4.3. Experimental results are shown in Section 4.4 and a detailed analysis of the hidden layers of our networks is provided in Section 4.5.

4.2 Related work

Given several tasks to perform on the same input data, instead of using one model per task, one can share parameters between models and tasks with the hypothesis that learning multiple tasks at the same time might improve the overall performance on each of the tasks [Caruana, 1998]. In a fully supervised setting, with annotated speech sequences that are also labelled with the identity of the speaker, multi-task learning has previously been used to improve the quality of ASR with an auxiliary speaker classification task [Pironkov et al., 2016], or such that each of both tasks (ASR and speaker identification) would benefit the other [Tang et al., 2016]. Performing both phonetic and speaker modelling in a weakly-supervised setting has been previously explored by Synnaeve and Dupoux [2014], on a small dataset. Here, we expand on this previous work in several ways. First, we scale up the architecture to deal with a considerably larger dataset [Panayotov et al., 2015]. Secondly, we improve on the loss function by adding a margin and investigating a triplet-based loss function as in [Kamper et al., 2015b]. We also use different types of input representations (either mel-filterbanks or the raw waveform) and measure the impact of this design choice on both phonetic and speaker modelling tasks. Finally we present out-of-domain experiments that show that the discriminability and invariance properties generalize to another English dataset and even another language.

While the main concept underlying Siamese networks is that signals of the same class should be close in the embedding space, and signals of different classes should be far in the embedding space, most classification or clustering algorithms do not require a particular level of absolute similarity between signals of the same classes. Rather, the classification of clustering accuracy depends on how similar signals of the same class are *relatively* to the similarity between signals of different classes. Thus, instead of considering a pair of signals, one can consider a triplet of signals, two belonging to the same category and one belonging to a different category. Ranking the similarity between the signals sharing the same category

and the signals of different categories allows computing a score that can be used to learn a representation that is discriminative for the task at hand. In speech applications, this has been previously used for acoustic modelling [Kamper et al., 2015b], and speaker modelling [Bredin, 2016].

4.3 Model

4.3.1 Input representations

Instead of forcing the network to encode the input into specific sub-word units (phonemes, diphones, triphones), we use the weakly supervised technique introduced in Chapter 2 which only specifies whether input sequences are utterances of same or different words, and if they are pronounced by same or different speakers, and let the network figure out by itself what are the most appropriate sub-word units. As detailed in Section 4.3.2, we evaluate the linguistic branch of our models in terms of phonetic discriminability of the embeddings that they learn and hence expect the networks to learn phone level representations even though only word level alignment is available as a ground truth information.

In the `MEL` setting, we represent the speech signal with mel-filterbanks frames with a window size of 25ms and a shift of 10ms. The networks learn phonetic and/or speaker embeddings of sub-words units, provided an input defined as a stack of 7 or 15 of successive filterbank frames, representing respectively a context of 70 and 150ms.

In the `WAV` setting, we train our architectures on the raw speech signal, sampled at 16kHz. A natural way of dealing with the raw waveform in a speech processing setting is to use convolutional neural networks. It allows learning several filters in the first layer, in an analogous way to hand-engineered filters that are used in standard speech features (mel-filterbanks, gammatone filters). In that setting, the networks learn phonetic and/or speaker embeddings of sub-words units, provided an input defined as a waveform segment representing an utterance of a particular word.

For training, we use annotations at the word level, but also the speaker identity. In the `MEL` setting, and in order to exploit pairs of identical words (same or different speakers), the two utterances are first realigned at the frame level using Dynamic Time Warping (DTW) [Sakoe and Chiba, 1978]. Sliding windows of stacked frames are then presented to the

Discriminability	A	B	X
Phonetic	/beg/ sp_1	/bag/ sp_1	/beg/ sp_2
Speaker	/beg/ sp_1	/beg/ sp_2	/bag/ sp_1

Table 4.1: Examples of A, B and X for both phonetic and speaker discriminability tasks. "sp_i" stands for speaker number i.

two entries of the siamese network. Dissimilar pairs are simply aligned along the diagonal. In the `WAV` setting, we face the additional difficulty that a DTW-type alignment on the raw waveform is less trivial than on standard speech features such as mel-filterbanks or MFCC. With an euclidean distance or a cosine distance, the DTW algorithm is linear in the number of dimensions and quadratic in the length of the sequences. This is acceptable when using features with a standard window stride (10ms), an utterance being represented by no more than a few hundreds frames. In the case of the waveform, the length of an utterance has an order of magnitude of thousands of values, making the cost of a DTW computation prohibitive. Moreover, an alignment obtained from a speech waveform would be an alignment on amplitude and would thus not be relevant for a speech modelling task, since linguistic content is characterized by the frequency information. This is why, in the `WAV` setting, instead of training our system on pre-aligned segments of waveform, we first align word utterances on mel-filterbanks, and then use the obtained path to align the output frames of the convolutional network that receives the full word waveform as entry. Figure 4-1 illustrates this architecture.

4.3.2 Discriminability and invariance

The objective of our training scheme is to "disentangle" phonetic content and speaker identity into separate representations. We want the phonetic embedding to show a good discriminability of phonetic classes and the speaker embedding to show a good discriminability of speaker classes. On the other hand, since we want to learn separate representations, all information related to speaker identity and independent from phonetic classes should be removed from the phonetic embedding, and conversely for the speaker embedding. This means that a speaker classification task performed on the phonetic embedding or a phonetic classification task on the speaker embedding should be as close as possible from chance level. This is what we refer to as invariance.

To evaluate the discriminability, but also the invariance properties of the embeddings learned by the system, we perform additional ABX discrimination tasks to the ones described in the previous chapters. In our experiments, A, B and X are triphones that may only differ by their central phoneme. When evaluating phonetic discriminability, A and B share the same speaker while their central phoneme is different, and X matches A on its phonetic content but is pronounced by a different speaker (the "across speaker" setting described in Chapter 2). Since we also want to evaluate speaker discriminability, we also use "across phoneme" ABX tasks. Switching B and X provides a speaker discriminability task *across* phonemes. Table 4.1 shows examples for both tasks.

Precisely, each triphone is represented as a stack of frames in the embedding space, and the distance between triphones is computed as the sum of the cosine distances between aligned frames after DTW. An ABX task is then performed per triplet and we show the average error over all triplets that can be found in the data.

We expect a phonetic embedding to show phonetic discriminability, which is characterized by a low ABX error rate on the *phone across speaker* task, and to show invariance to speaker identity, which is characterized by an ABX error as close as possible from 50% on the *speaker across phone* task. Conversely, better speaker embeddings have lower *speaker across phone* error rate, and a high *phone across speaker* error.

4.3.3 Multi-task siamese network

A single-task siamese architecture is trained using labeled pairs $(x, x', y^{phn}, y^{spk})$ where x and x' are two input speech segments, $y^{phn} \in \{0, 1\}$ is 1 if x and x' are phonetically similar and $y^{spk} \in \{0, 1\}$ is 1 if x and x' are said by the same speaker. Given x, the network outputs an embedding $\mathbf{e}(x) \in R^d$.

As in Chapter 3, Siamese networks are trained using a loss function defined on pairs which, given any two embeddings $\mathbf{e}$, $\mathbf{e}'$ in R^d and a label $y \in \{0, 1\}$, enforces that $\mathbf{e}$ should be close to $\mathbf{e}'$ if $y = 1$, while the two embeddings should be far away if $y = 0$. However, in the case where $y = 0$ we replace the $\cos^2$ function by a cosine with margin. The pairwise

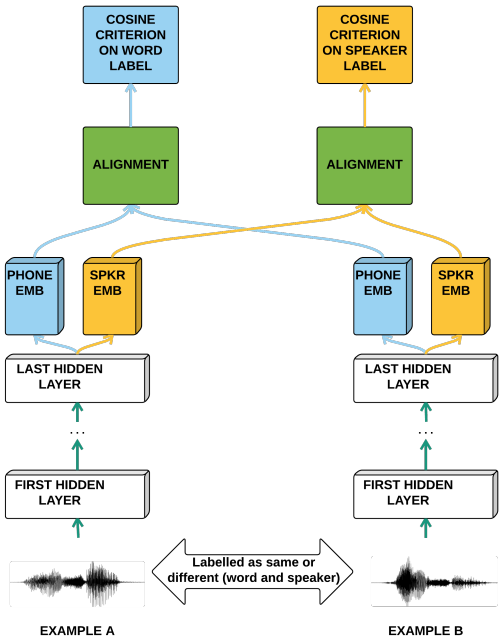


Figure 4-1: A multi-ouput siamese network in the `WAV` setting. All parameters of each of the two branches at a given depth are shared.

loss function we use is

$$\ell\gamma(\mathbf{e}, \mathbf{e}', y) = \begin{cases} -\cos\left(\mathbf{e}, \mathbf{e}'\right) & \text{if } y = 1 \\ \max(0, \cos(\mathbf{e}, \mathbf{e}') - \gamma) & \text{if } y = 0 \end{cases}, \tag{4.1}$$

where γ is a margin hyperparameter. Hence, when two inputs have a "same" label, this loss enforces co-linearity of their embeddings. Conversely, if the inputs are "different", the loss enforces that their cosine similarity is lower than γ. Providing the loss function with y^{phn} as a label allows learning a phonetic embedding, while using y^{spk} allows learning a speaker embedding. Hence the loss function for a phonetic modelling siamese network is $\ell\gamma(\mathbf{e}, \mathbf{e}', y^{phn})$ while the loss used to train a speaker modelling siamese network is $\ell\gamma(\mathbf{e}, \mathbf{e}', y^{spk})$. When learning multiple embeddings, we may have different margin parameters γ^{phn} and γ^{spk} for phonetic and speaker embeddings respectively. The loss of the multi-output network is then

$$L(x, x', y^{phn}, y^{spk}) = \ell_{\gamma^{phn}}(\mathbf{e}^{phn}(x), \mathbf{e}^{phn}(x'), y^{phn}) \tag{4.2}$$

$$+ \ell_{\gamma^{spk}}(\mathbf{e}^{spk}(x), \mathbf{e}^{spk}(x'), y^{spk})\,. \tag{4.3}$$

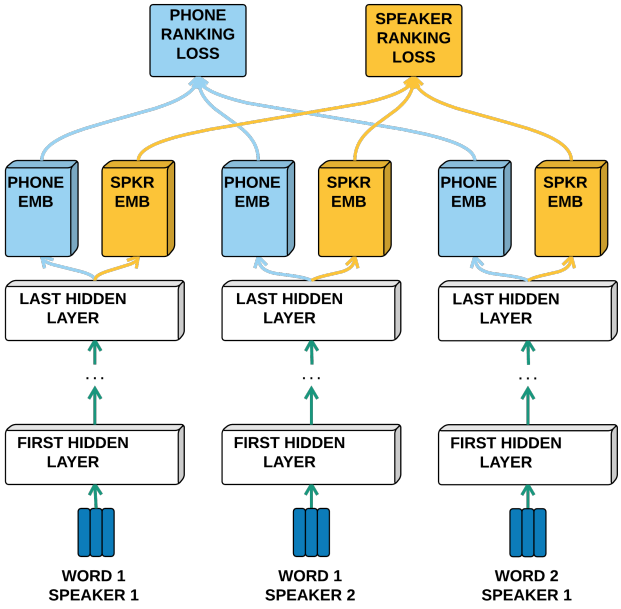


Figure 4-2: A multi-ouput triamese network in the MEL setting. All parameters of each of the three branches at a given depth are shared.

The absence of weighting factors in the summation of the phonetic and speaker loss means that we give an equal importance to each of the task. A multi-output siamese network is shown in Fig. 4-1.

4.3.4 Multi-task triamese network

The triamese network uses a triplet-based loss function [Bredin, 2016, Chechik et al., 2010, Kamper et al., 2015b, Van Der Maaten and Weinberger, 2012]. The model has the same architecture as before, but now the data takes the form (x_1^1, x_1^2, x_2^1) where (x_1^1, x_1^2) are speech segments with similar phonetic content from two different speakers, and (x_1^1, x_2^1) are segments from two different words said by the same speaker.

A triplet loss enforces constraints on relative similarities between pairs. For phonetic embeddings $\mathbf{e}^{phn}$, two utterances of the same word pronounced by different speakers (x_1^1, x_1^2) should be more similar than utterances of different words pronounced by the same speaker (x_1^1, x_2^1). The rule is inverted for speaker embeddings: two different words pronounced by the same speaker should be closer in embedding space than two utterances of the same

word pronounced by different speakers. This setting is not exhaustive regarding the types of triplets we could use to learn our tasks. For instance, a triplet of the type (x_1^2, x_1^2, x_2^1) could be used to learn both embeddings by making $(x_1^2,\ x_1^2)$ closer than $(x_1^2,\ x_2^1)$ for both tasks. However, since x_1^2 and x_2^1 are different regarding both factors of variation, this task is simpler than the previous one. In a context in which we would have to select triplets in a vast quantity of possible ones, focusing solely on the hardest triplets, and thus the most informative ones, is an efficient solution that we choose to adopt. Formally, the triplet loss is defined for any three embeddings $\mathbf{e}, \mathbf{e}', \mathbf{e}''$ as:

$$\tilde{\ell}_\gamma(\mathbf{e}, \mathbf{e}', \mathbf{e}'') = \max\left(0, \gamma - \cos(\mathbf{e}, \mathbf{e}') + \cos(\mathbf{e}, \mathbf{e}'')\right).$$

The losses for each embeddings are then:

$$\tilde{\ell}^{phn}(x_1^1, x_1^2, x_2^1) = \tilde{\ell}_{\gamma^{phn}}\left(\mathbf{e}^{phn}(x_1^1), \mathbf{e}^{phn}(x_1^2), \mathbf{e}^{phn}(x_2^1)\right),$$

$$\tilde{\ell}^{spk}(x_1^1, x_1^2, x_2^1) = \tilde{\ell}_{\gamma^{spk}}\left(\mathbf{e}^{spk}(x_1^1), \mathbf{e}^{spk}(x_2^1), \mathbf{e}^{spk}(x_1^2)\right).$$

For the multi-output network, the final loss is $\tilde{\ell}^{phn} + \tilde{\ell}^{spk}$. A multi-output triamese is shown in Fig. 4-2.

4.4 Experimental setup

We train both siamese and triamese networks for three types of task (joint learning, phonetic modelling only, speaker modelling only), resulting in six different training schemes.

The neural networks are trained on the 360 hours of read speech (920 speakers) constituting the *train_clean_360* subset of the Librispeech dataset [Panayotov et al., 2015]. We obtained the word-level annotations by force-aligning a state-of-the-art HMM-DNN [Panayotov et al., 2015] with transcriptions at the phone level, and then segmented the speech utterances at word boundaries.

In the `MEL` setting, after preliminary experiments, we focused on a fully connected deep neural net architecture with four hidden layers with 1000 units and a final embedding layer of size $d = 100$. We used as the activation function the RReLU non-linearity [Xu et al., 2015] at each layer (ReLUs exhibited similar performances).

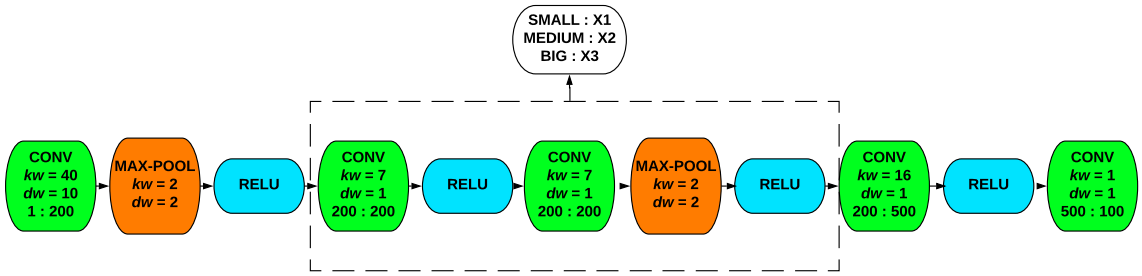


Figure 4-3: Our convolutional architecture trained on the waveform. The part surrounded by the dashed rectangle is repeated 1, 2 and 3 times respectively for the Small, Medium and Big architectures.

In the `WAV` setting, we used convolutional architectures composed of three types of layers: 1-d convolutions, max-pooling and ReLU. We trained three types of networks, that differ by their depth and their total spread (the time context that is used to generate an output frame). In the remainder of this chapter these networks are referred as Small, Medium, and Big, and their respective depth and spread parameters are as follows:

- Small: 5 convolutional layers, spread of 57ms
- Medium: 7 convolutional layers, spread of 127ms
- Big: 9 convolutional layers, spread of 267ms

These three architectures have the same structure for their input and output layers, they only differ by the central layers. A visualization of these architectures, with their detailed structure, can be seen in Fig. 4-3.

In the `MEL` setting, we use Adadelta [Zeiler, 2012] with interpolation parameter 0.9 and epsilon 10^{-6} to train the siamese architecture, whereas plain stochastic gradient descent (SGD) performs slightly better for the triamese model. In the `WAV` setting, we only use SGD as our optimization algorithm. In both `MEL` and `WAV` settings, the learning rate for SGD starts at 0.01 and is halved when the error on the development set stops decreasing (with a minimum of 10^{-6}). In the `MEL` setting, the margin parameters (γ^{phn} and γ^{spk}), the weight decay, and the number of frames in an input stack are respectively chosen among $\{0.15, 0.5, 0.85\}$, $\{0, 0.001\}$ and $\{7, 15\}$. In the `WAV` setting, we do not use weight decay,

and since the results of `MEL` suggested that a margin of 0.15 was significantly worse we only cross-validate the margin hyperparameters (γ^{phn} and γ^{spk}) among $\{0.5, 0.85\}$. We use the *dev_clean* split of the dataset for early stopping and cross-validation of the margin hyperparameters.

4.4.1 Evaluation setups

In-domain evaluation

Evaluations on the Librispeech dataset are computed on the *test_clean* subset. We use annotations at the phoneme level from the forced alignment to extract all relevant triplets from the test set. We then subsample randomly 10% of the triplets to get 600k ABX triplets for the evaluation, from 40 speakers.

Out-of-domain evaluation

In order to evaluate the robustness of the learned representation across datasets and languages, we also perform two sets of out-of-domain experiments. First, we evaluate our embeddings on the training set of the TIMIT dataset [Garofolo et al., 1993]. We extract all triplets from the train set of the standard train/dev/test split. We then subsample randomly 10% of these triplets, and obtain 1.87m ABX triplets total, with 462 speakers.

We also evaluate out-of-domain performance across languages by testing our embeddings on the Xitsonga dialect, on a subset of the NCHLT corpus [Vries et al., 2014]. This corpus was used in the zerospeech 2015 challenge [Versteegh et al., 2015], and we will compare our method to the best in-domain unsupervised system. The corpus used for evaluation contains 240k ABX triplets, for 24 speakers.

4.4.2 In-Domain Results

We present the ABX error rate of phone and/or speaker embeddings, each one on both *phone across speaker* and *speaker across phone* task. Table 4.2 reports ABX error rates on Librispeech on the speech features. The evaluation tasks are either ABX on phones across speakers (phn) or ABX on speakers across phones (spk). "Siamese" and "Triamese" are followed by the number of frames in an input stack. "Single" means that the phonetic and

model	task	phone embed.		speaker embed.	
		phn	spk	phn	spk
MEL7	-	24.5	32.9	24.5	32.9
Siamese7	single	10.9	46.0	46.4	23.9
	double	10.5	45.9	45.4	9.3
Siamese15	single	9.7	47.1	45.8	12.4
	double	10.2	46.6	45.3	8.7
Triamese7	single	10.0	46.0	45.0	10.0
	double	11.5	45.0	45.7	9.4
Triamese15	single	9.8	46.9	44.6	9.4
	double	10.7	46.2	44.7	8.1

Table 4.2: In-domain ABX error rates (%) in the `MEL` setting.

speaker embeddings were trained separately in single output networks, whereas "double" refers to a multi-task network. As a baseline, we also report the ABX errors on stacks of 7 mel-filterbanks frames, referred as MEL7, which were shown to give good results on TIMIT in Chapter 3. In that case, the raw stack of mel-filterbanks is used both as direct phonetic and speaker embeddings. Table 4.3 reports ABX error rates on Librispeech of models trained on the raw waveform. The notations are the same, except that "small", "medium", and "big" refer to the depth of the convolutional network.

In the `MEL` setting, and for all networks, the phone and speaker tasks show high discriminability on their respective embeddings with an error rate around 10% (best score, respectively of 9.7% and 8.7%). At the same time, the scores on the mismatched embeddings (phonetic embedding for a speaker task and speaker embedding for a phonetic task) are within 7% of the chance level. This means that the embeddings have learned to be discriminative for their relevant task but also not to code the irrelevant one. This contrasts with the MEL7 input representations that encode both dimensions, as they are low-level features intended to replicate perceptual representations for both speech and speaker recognition. Moreover, even though comparisons are limited because the datasets are different, we achieve here a level of disentanglement that was not reached by Synnaeve and Dupoux [2014], in which phonetic embeddings had phonetic discriminability close to the mel-filterbanks (30.4% and 34.1% error respectively), and were less speaker-invariant than mel-filterbanks (30.8% and 38% error respectively).

In addition, we can see that the double embedding architectures do roughly as well as the single ones, even though the former have to share most of the network's weights for the two

model	network type	task	phone embed.		speaker embed.	
			phn	spk	phn	spk
MEL7	-	-	24.5	32.9	24.5	32.9
Siamese	small	single	15.7	44.1	44.9	19.1
		double	15.9	44	43.8	15.1
	medium	single	14	46.8	45.1	13.6
		double	14.5	46.5	44.7	10.1
	big	single	19.8	48.1	45.1	10.1
		double	20.5	48.2	44.7	7
Triamese	small	single	14	44.6	45.5	7.3
		double	14.7	43.5	45.8	6.7
	medium	single	14.1	46.6	45.2	6.5
		double	15.6	45.4	45.5	6
	big	single	20.4	47.7	44.9	6.3
		double	22.3	46.8	45	5.6

Table 4.3: In-domain ABX error rates (%) in the `WAV` setting.

competing tasks. The speaker embedding (tested on the speaker task) seems to consistently benefit from the double training regime compared to a network trained only on a single task, these gains ranging from 0.6% to 14.5% (absolute). The phone tasks, in contrast, are less consistently affected, some architectures showing a small gain and most others a small cost. Finally, while there is no strong difference between the performance of a Siamese and a Triamese network on the same task, the latter seems to bring some improvement to the quality of speaker embeddings

In the `WAV` setting, the performance is more task-dependent. For the phonetic task, depth helps from Small to Medium but then degrades as the architectures get deeper. As in the `MEL` setting, we also observe a small increase in ABX error when comparing a single-task architecture with its multi-task equivalent. The lowest ABX error is at 14%, which is significantly worse than the best phonetic embedding of the `MEL` setting (9.7%). This corroborates the observation made in Section 1.3 that mel-filterbanks are a very strong baseline for speech recognition. On the other hand, the quality of speaker embeddings consistently improves with depth, and still benefits of multi-task learning and using a Triamese architecture. The best models show a performance that is significantly higher than the best `MEL` model, with 5.6% error against 8.1%. It is not trivial to compare the architectures between the `WAV` and the `MEL` Settings, as one is fully connected and the other is convolutional, but in terms of number of parameters and the time duration used to model a speaker, Table 4.4 shows

model	nb parameters	spread (ms)	ABX error (%)
Best	3.39M	267	5.6
Best small	2.27M	57	6.7
Best `MEL`	3.38M	165	8.1

Table 4.4: Number of parameters, size of time context (in miliseconds) and *speaker across phone* ABX error (%) for the best model from the `WAV` setting, the best Small model and the best model from the `MEL` setting.

	phone embed.		speaker embed.	
model	phn	spk	phn	spk
Language: English (TIMIT)				
MEL7	20.5	39.7	20.5	39.7
DNN supervised	9.2			
Best ScatABnet	9.8			
Tri15 (double)	10.3	47.9	43.4	14.2
Tri15 (single)	9.2	48.7		
Best wav overall	12.7	45.9	38.4	25.8
Best wav phone	13.5	47.5		
Language: Xitsonga (NCHLT)				
MEL7	30.1	25.8	30.1	25.8
Best ScatABnet	15.8			
Tri15 (double)	15.4	41.6	44.7	14.3
Tri15 (single)	15.5	42.6		
Best wav overall	20.5	38.1	42.5	12
Best wav phone	20.2	42.1		

Table 4.5: Out-of-domain ABX error rates (%) in the `MEL` and `WAV` settings.

that even with a spread that is three times lower and a number of parameters that is 33% lower, a `WAV` model still performs significantly better than the best `MEL` model on the speaker modelling task, with an ABX error of 6.7% against 8.1%.

4.4.3 Out-of-domain results

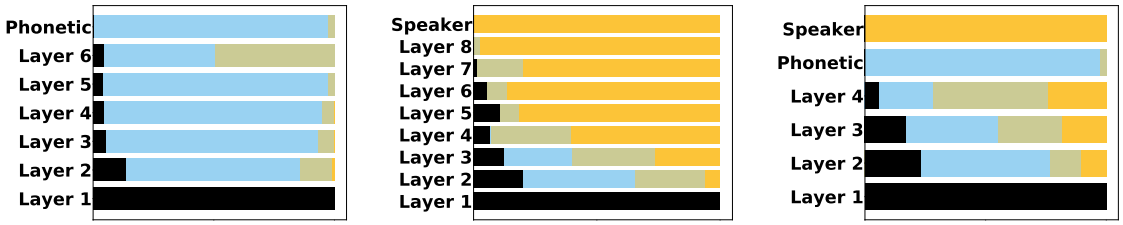
Table 4.5 reports out-of-domain ABX results on a different dataset in English (TIMIT) and on a different language, Xitsonga. The results for Tri15 are obtained from extracting output embeddings from a (single or double) Triamese neural network previously trained on Librispeech. MEL7 is an untrained stacked filterbank baseline, and ScatABnet is the best model from Chapter 3, either state-of-the art model for a weakly supervised siamese architecture trained on the TIMIT or an unsupervised architecture trained on the Xitsonga

dataset (respectively) using a deep scattering spectrum as input. For TIMIT, the DNN "topline" is the output of a supervised neural network trained as a phone classifier on the TIMIT train set [Synnaeve et al., 2014]. For reference, we show the same MEL7 baseline as in the in-domain experiments.

In the `MEL` setting, models trained on Librispeech generalize well to TIMIT, showing that the robustness of the representations. For both phonetic and speaker modelling tasks, the models keep very good levels of discriminability and invariance on this new dataset. This is particularly remarkable for the speaker modelling task, since the TIMIT train set contains 462 speakers, while there were only 40 speakers in the in-domain evaluation. Moreover, models trained on English (Librispeech) generalize to Xitsonga, a language typologically unrelated to English, containing a large array of consonants (54) including some click consonants and a contrast between breathy and modal voiced consonants which is totally absent in English. Third, these out-of-domain models happen to beat the previous in-domain state-of-the-art that used a similar training scheme in terms of architectures and loss function. On TIMIT, the single output triamese network trained on pairs of words has a *phone across speaker* ABX of 9.2%, which is equivalent to the in-domain supervised phone classifier DNN.

In the `WAV` setting, we can observe that the transfer to another dataset does not exacerbate the gap in performance for the phonetic task, when compared to the `MEL` architectures. The architectures trained on the waveform seem to generalize as well as the architectures trained on features, showing an ABX error on TIMIT that is lower than the error on the in-domain test-set. However, the speaker modelling task is considerably more impacted by the transfer, and the `WAV` architecture shows a far poorer transfer than the `MEL` architecture (25.8% ABX error, compared to 14.2%), even though the `WAV` models perform significantly better on the speaker task in-domain.

When testing `WAV` architectures on Xitsonga, we can see that the results on the phonetic task are coherent with the gap in performance we observed with the `MEL` architectures on both Librispeech and TIMIT. However, we can observe that the architecture of the `WAV` setting generalizes significantly better to the Xitsonga than the best performing architecture of the `MEL` setting.



(a) Phonetic network (b) Speaker network (c) Multi-task network

Figure 4-4: Visualizations of what layers code through the network. The left barchart is a phonetic modelling network, the middle barchart is a speaker modelling network, and the right one is a multi-task network. In blue are units that code for the phonetic information, in yellow, the speaker information, in green the units that code for both, and in black the units that code for none.

4.5 Detailed analysis of discriminability and invariance in the networks

In order to get a better understanding of what kind of information is encoded throughout our convolutional networks trained on the waveform, we perform an ANOVA (analysis of variance) [Fisher, 1925]. For each unit of each convolutional layer in our network, we compute both its intra-class and inter-class variability along all sentences from the test set, for both phonetic and speaker modelling. For instance, for the phonetic task, the intra-class variability of a unit refers to the variance of its activations along all utterances of a same phoneme. Its inter-class variability is computed as the variance of its intra-class means along all phonetic classes. Taking the ratio of the inter-class variance and the intra-class variance gives a score that indicates how much a unit codes for a particular task. The higher the score, the more the unit reacts to a change of class for that task. Our boundaries to decide for which task(s) a unit codes are defined as the median score for all units over all layers, for each task. Then, for a given unit, if phonetic and speaker scores are respectively above the phonetic and speaker median, the unit codes for both task. If only the phonetic score is above its median, the unit codes for the phone, and conversely for the speaker. Finally, if both scores are below the median, the unit is considered as coding for nothing.

Fig. 4-4 shows the results of this analysis on three networks trained in the `WAV` setting: the first one is our best single-task architecture for the phonetic modelling, the second one is the best single-task architecture for the speaker modelling, and finally the last one is

the best multi-task architecture (in the sense that the average of its phonetic and speaker ABX errors is the lowest). The various statistics (intra-class and inter-class variability) are computed across these three architectures jointly. First, we can see that in each of the three architectures, the first layer codes for nothing. We can interpret it in the sense that the waveform is such a raw representation of speech that the first layer of the network might systematically perform low-level processing that can not directly be linked to speech or speaker modelling. In higher layers, differences between models emerge. For the single-task phonetic network, we can see that the speaker information is removed as soon as the second layer, and the final output almost purely models phonetic classes. On the other hand, single-task speaker network keeps phonetic information up to several layers and then outputs an embedding that purely encodes speaker information (100 units out of 100). This asymmetry between tasks can be related to the successful use of posteriors from neural networks trained for ASR to compute i-vectors for speaker identification [Lei et al., 2014, Sadjadi et al., 2016]. As i-vectors are used to model a speaker by shifts in phonetic classes, it is possible that our neural network learns speaker representations from phonetic information. This can explain why multi-task learning benefits the speaker modelling task and does not benefit the phonetic modelling task: phonetic features are useful for learning good speaker representations, while the usefulness of speaker related information for phonetic modelling is less obvious according to our analysis. Finally, the analysis of the multi-task network shows that as we go deeper, the layers model both tasks more and more, and that both output embeddings are very pure for their task.

4.6 Conclusion

We have demonstrated that a siamese or triamese architecture, together with a weak supervision using only same-different information regarding word and speaker identity can learn embeddings that are very selective in one dimension and invariant in the other: indeed, our best embeddings showed around a $6-10\%$ error rate in one task and near chance in the other. Moreover, we showed that it was possible to learn these two orthogonal embeddings from one network, and without speech features, thereby demonstrating effective disentanglement of phoneme and speaker information from the raw waveform. Finally, we showed that these disentangling networks could generalize to out-of-domain datasets (a different

English dataset, and a different, under-resourced, language), even beating the in-domain state-of-the-art. This shows that transfer learning is a promising avenue for exploiting data from high-resource languages to benefit low-resource ones.

In detail, the multi-task networks differed somewhat from the single-task ones. In particular, whereas the speaker task benefited consistently from the joint training, this was not the case for the phone task. A thorough ANOVA study on our trained systems showed that a model trained for speaker modelling observed would model the phonetic content up to high-level layers, while a model trained for phonetic modelling would remove speaker information in early layers. Both the ABX errors and the ANOVA corroborate the previous observation that speaker identification systems improve their performance by using side phonetic/linguistic information while the benefit of adding speaker information to state of the art ASR has been more elusive.

We also compared models trained on the waveform and on mel-filterbanks. We observed that even though models trained on the waveform produced significantly worse phonetic embeddings, they were particularly adapted to speaker modelling, yielding excellent embeddings even with a context size as small as 57ms, while speaker verification systems are typically trained on segments of hundreds to thousands of miliseconds. Adding a weighting factor to the speaker loss, relatively to the phonetic loss, could rebalance the overall performance of the system over the two tasks, and improve the phonetic discriminability of the learnt representation. Our findings are consistent with the hypothesis we formulated in Section 1.4: mel-filterbanks are likely biased in favor of speech recognition rather than paralinguistic tasks (including speaker identification). This is also supported by the higher phonetic discriminability of the MEL7 baseline (24.5%) than speaker discriminability (32.9%).

However, these experiments lack the sufficient rigor to make a definitive conclusion on which input representation, of the waveform or the mel-filterbanks, is more adapted to one task or the other. Indeed, even though we control for the number of parameters in our neural networks, their architectures (convolutional or fully connected, choice of activation function, number of layers, etc.) are too different to guarantee that the differences in performance between the `WAV` and `MEL` settings are only due to the input representation, or are artifacts of the aforementioned confounding factors. This uncertainty also holds for the experiments of Chapter 3, as the higher dimension of the deep scattering spectrum (relatively to mel-

filterbanks) naturally increases the expressivity of the Siamese network trained on top of this representation. Finally, we are not satisfied with the gap in quality of phoneme embeddings between the `WAV` and `MEL` settings, and will thus explore furthermore speech recognition from the raw waveform in the next sections.

Part III

Learning the speech front-end for speech recognition and paralinguistic tasks

I would ask, why use the Mel scale now, since it appears to be biased? If anyone wants a Mel scale they should do it over, controlling carefully for order bias and using plenty of subjects − more than in the past − and using both musicians and non-musicians to search for any differences in performance that may be governed by musician/non-musician differences or subject differences generally.

Donald D. Greenwood

Chapter 5

Time-Domain filterbanks: a learnable frontend of speech

This chapter is based on Learning Filterbanks from Raw Speech for Phone Recognition [Zeghidour et al., 2017], accepted for poster presentation at ICASSP 2018, and a joint work with Nicolas Usunier, Iasonas Kokkinos, Thomas Schatz, Gabriel Synnaeve and Emmanuel Dupoux.

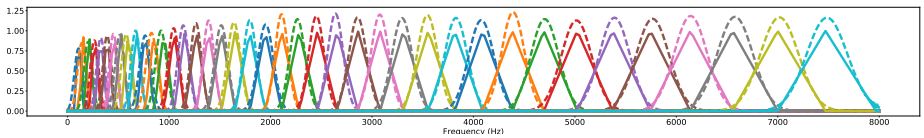
5.1 Introduction

In Chapter 3, we conducted a comparative study of speech features on phonetic modelling and showed that this design choice had a significant impact on the performance of the final system. To circumvent this selection step, as well as inherent limitations of fixed features exposed in Section 1.2, we then decided to experiments with models that can learn their low-level representations directly from the raw waveform, jointly with the main architecture. These encouraging results lead us to keep exploring the idea of replacing mel-filterbanks by neural layers, however the limitations of the models and experimental protocol developed in Chapter 4 also incite us to add more constraints to our learnable alternative. First, it should have an equivalent structure to mel-filterbanks (number of filters, capacity, context size, context stride) in order to compare the learnt and fixed approaches without any confounding factor. Moreover, the gap in quality between phonetic embeddings learned from mel-filterbanks or the waveform suggests that we should carefully study the computation of

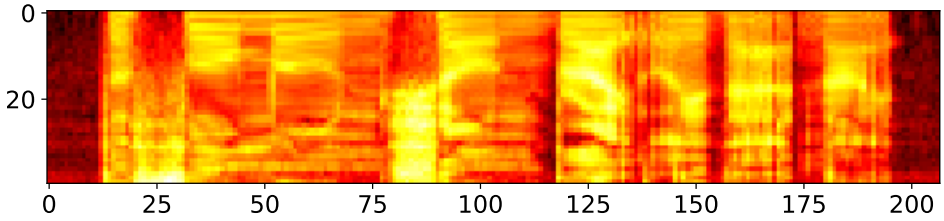
mel-filterbanks to guarantee that their essential components are integrated into our neural frontend. These two insights drive the design of our learnable architecture, the Time-Domain filterbanks, that we introduce in this Chapter.

Sharing this idea of taking inspiration from the computation of mel-filterbanks, Tjandra et al. [2017a] pre-train 9 layers of convolutions (including Network-in-Network layers [Lin et al., 2013]) to reconstruct mel-filterbanks, and then use them as the lower layers of an end-to-end sequence-to-sequence system. This allows outperforming mel-filterbanks on a speech recognition task, however these experiments suffer from the same confounding factors as our experiments in Chapter 4. Indeed, even though these layers of convolutions are pre-trained to reconstruct mel-filterbanks, they are then fine-tuned with the rest of the network, to which they add a lot of capacity. This explains how this approach outperforms mel-filterbanks, as mel-filterbanks involve way fewer operations than these convolutional layers.

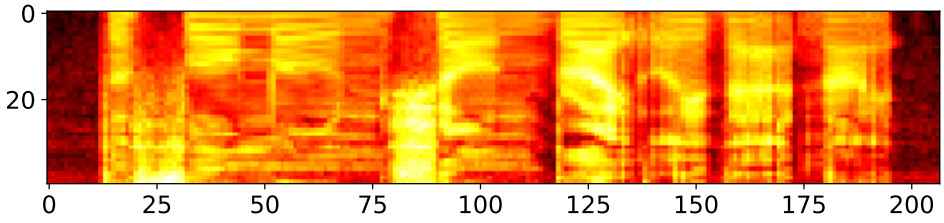
We also focus on mel-filterbanks because they are the front-end of state-of-the-art phone [Tóth, 2015] and speech [Han et al., 2017, Xiong et al., 2016] recognition systems. As in Chapter 3, we build on [Andén and Mallat, 2014]. However, in that Part we are not interested in the derivation of a deep scattering spectrum, but rather on the first-order scattering transform. Indeed, in Section 3.3, we briefly mention that a scattering transform approximates mel-filterbanks in time-domain when it is defined with an appropriate bank of filters. By constructing a neural network that computes a scattering transform, the Time-Domain filterbanks, we adress all the aforementioned limitations of our previous approach. First, a scattering transform moves the standard computation of mel-filterbanks to the time-domain, and thus works directly on the waveform. Moreover, as shown in the next Section, the number of operations, as well as the structure of a scattering transform is equivalent to mel-filterbanks, which addresses the capacity problem. Finally, initializing our neural layers with an actual scattering transform allows us to approximate mel-filterbanks before training, while being able to fine-tune the frontend during training. This leads us to study an architecture using a convolutional layer with complex-valued weights, followed by a modulus operator and a low-pass filter. In contrast to [Tjandra et al., 2017a], Time-Domain filterbanks are a lightweight architecture that serves as a plug-in, learnable replacement to mel-filterbanks in deep neural networks. Moreover, we avoid pretraining by initializing the complex convolution weights with Gabor wavelets whose center frequency and bandwidth match those of mel-filterbanks.



(a) Mel-filterbanks (solid) and Gabor filter approximation (dashed)



(b) Mel-filterbanks



(c) Time-Domain filterbanks with the Gabor filters

Figure 5-1: Frequency response of filters, and output of mel-filterbanks and their time-domain approximation on a sentence of TIMIT.

In the next Section, we derive the Time-Domain filterbanks, in terms of network architecture, filters specifications, and choice of initialization. Based on the observation, made in sections 1.2 and 1.3, that Automatic Speech Recognition (ASR) could benefit from learning from the waveform, we perform phone recognition experiments on TIMIT as a proof of concept for Time-Domain filterbanks. We show that given competitive end-to-end models trained with mel-filterbanks as inputs, training the same systems but just replacing the mel-filterbanks with the learnable architecture leads to performances that are consistently better than when using mel-filterbanks.

5.2 Time-Domain filterbanks

We present the standard mel-filterbanks and their practical implementation. We then describe a learnable replacement of mel-filterbanks that uses only convolution operations in time domain, and how to set the weights to reproduce mel-filterbanks.

5.2.1 Mel-filterbanks computation

Given an input signal x, mel-filterbanks are computed by first taking the short-time Fourier transform (STFT) of x followed by taking averages in the frequency domain according to triangular filters with centered frequency and bandwidth that increase linearly in log-scale. More formally, let ϕ be a Hann window of width s and $(\psi_n)_{n=1..N}$ be N filters whose squared frequency response are triangles centered on $(\eta_n)_{n=1..N}$ with full width at half maximum (FWHM) $(w_n)_{n=1..N}$. Denoting by $x_t : u \mapsto x(u)\phi(t-u)$ the windowed signal at time step t, and $\hat{f}$ the Fourier transform of function f, the filterbank is the set of N functions $(t \mapsto Mx(t,n))_{n=1..N}$:

$$Mx(t,n) = \frac{1}{2\pi} \int |\hat{x}_t(\omega)|^2 |\hat{\psi}_n(\omega)|^2 d\omega \,, \tag{5.1}$$

One can observe that this formula does not include the log-compression. This is due to the fact that we can simply include a log non-linearity of our learnable frontend to replicate it, so there is no need of approximating this operation.

5.2.2 Approximating mel-filterbanks with convolutions in time

Andén and Mallat [2014] show that these standard mel-filterbanks computed in the frequency domain can be approximated with a first order scattering transform (see Section II of [Andén and Mallat, 2014] for a proof):

$$Mx(t,n) \approx |x * \varphi_n|^2 * |\phi|^2(t). \tag{5.2}$$

where φ_n is a wavelet that approximates the n-th triangular filter in frequency, i.e. $|\hat{\varphi}_n|^2 \approx |\hat{\psi}_n|^2$, while $\phi(t)$ is the Hann window also used for the mel-filterbanks. The approximation is valid when the time support of φ_n is smaller than that of ϕ.

As described in Section 3.3, this approximation is the foundation of the deep scattering spectrum [Andén and Mallat, 2014], which cascades scattering transforms to retrieve information that is lost in the mel-filterbanks. However, in this part, we do not use the deep scattering spectrum. Rather, first-order scattering coefficients provide us with both a design for the first layers of the network architecture to operate on the waveform, and an initialization that approximates mel-filterbanks.

Given the mel-filterbanks center frequencies $(\eta_n)_{n=1..N}$ and FWHM $(w_n)_{n=1..N}$, we use (5.2) to approximate mel-filterbanks with Gabor wavelets:

$$\varphi_n(t) \propto e^{-2\pi i \xi_n t} \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{t^2}{2\sigma_n^2}}. \tag{5.3}$$

Each φ_n is a Gaussian $\mathcal{N}(0, \sigma_n)$ modulated by a sinusoid of frequency ξ_n. In the Fourier domain this product becomes a convolution between a Gaussian $\mathcal{N}(0, \sigma_n^{-1})$ and a Dirac in ξ_n, which yields a Gaussian $\mathcal{N}(\xi_n, \sigma_n^{-1})$. We can thus choose ξ_n and σ_n such that the frequency response of the Gabor φ_n matches the corresponding mel-filterbank (approximating a triangular response by a Gaussian one). Since φ_n is centered in ξ_n, and we want it to match a mel-filter centered in η_n, we simply choose $\xi_n = \eta_n$. Similarly, we want to set the width parameter σ_n of the Gabor wavelet to match the desired FWHM w_n. The FWHM of a Gaussian $\mathcal{N}(\xi_n, \sigma_n^{-1})$ is equal to $2\sqrt{2\log 2}\sigma_n^{-1}$, so to have $2\sqrt{2\log 2}\sigma_n^{-1} = w_n$, we take $\sigma_n = \frac{2\sqrt{2\log 2}}{w_n}$.

Each φ_n is then normalized to have the same energy as ψ_n. Figure 5-1 (a) shows in frequency-domain the triangular averaging operators of usual mel-filterbanks and the corresponding Gabor wavelets. Figures 5-1 (b) and (c) compare the 40-dimensional spectrograms of the mel-filterbanks and the Gabor wavelet approximation on a random sentence of the TIMIT corpus after mean-variance normalization, showing that the spectrograms are similar.

Mel-filterbanks specification. The standard setting in speech recognition is to start from the waveform sampled at 16kHz and represented as 16-bit signed integers. The STFT is computed with 512 frequency bins using Hann windows of width 25ms, and decimation is applied by taking the STFT every 10ms. There are $N = 40$ filters, with center frequencies $(\eta_n)_{n=1..N}$ that span the range $64Hz - 8000Hz$ by being equally spaced on a mel-scale. The final features are the $\log(\max(Mx(t, n), 1))$. In practice, the STFT is applied to the raw signal after a pre-emphasis with parameter 0.97, and coefficients have mean-variance normalization per utterance.

Learnable architecture specification. The time-domain convolutional architecture is summarized in Table 5.1. With a waveform sampled at 16kHz, a Hann window is a convolution operator with a span of $W = 400$ samples (25ms). Since the energy of the Gabor wavelets approximating standard mel-filterbanks has a time spread smaller than the Hann

Layer type	Input size	Output size	Width	Stride
Conv.	1	80	400	1
L2-Pooling	80	40	-	-
Square	-	-	-	-
Grouped conv.	40	40	400	160
Absolute value	-	-	-	-
Add 1, Log	-	-	-	-

Table 5.1: Details of the layers for the Time-Domain filterbanks.

window, the complex wavelet+modulus operations $|x * \varphi_n|^2$ are implemented as a convolutional layer taking the raw wav as input, with a width $W = 400$ and $2N = 80$ filters (40 filters for the real and imaginary parts respectively). This layer is on the top row of Table 5.1. The modulus operator is implemented with "feature L2 pooling", a layer taking an input z of size $2N$ and outputs z' of size N such that $z'_k = \sqrt{z^2_{2k-1} + z^2_{2k}}$. The windowing layer (third row of Table 5.1) is a grouped convolution, meaning that each output filter only sees the input filter with the same index. The decimation of 10ms is implemented in the stride of 160 of this layer. Notice that to approximate the mel-filterbanks, the square of the Hann window is used and biases in both convolutional layers are set to zero. We keep them to zero during training. We add a log compression to the output of the grouped convolution after adding 1 to its absolute value since we do not have positivity constraints on the weights when learning. Contrarily to the mel-filterbanks, there is no mean-variance normalization after the convolutions, but on the waveform. In the default implementation of the Time-Domain filterbanks, we do not apply pre-emphasis. However, in our last experiment, we add a convolutional layer below the Time-Domain filterbanks, with width 2 and stride 1, initialized with the pre-emphasis parameters, as another learnable component.

5.3 Experiments

5.3.1 Setting

We perform phone recognition experiments on TIMIT [Garofolo et al., 1993] using the standard train/dev/test split. We train and evaluate our models with 39 phonemes, following the protocol of Lee and Hon [1988]. We experiment with three architectures. The first one

Learning mode	Dev PER	Test PER
mel-filterbanks	17.8	20.6
Fixed	18.3	21.8
Learn-all	17.4	20.6
Learn-filterbank	17.3	20.3
Randinit	29.2	31.7

Table 5.2: PER of the CNN-5L-ReLU-do0.7 model trained on mel-filterbanks and different learning setups of Time-Domain filterbanks.

consists of 5 layers of convolution of width 5 and 1000 feature maps, with ReLU activation functions, and a dropout [Hinton et al., 2012b] of 0.5 on every layer but the input and output ones. The second model has the same architecture but a dropout of 0.7 is used. The third model has 8 layers of convolution, PReLU [He et al., 2015] nonlinearities and a dropout of 0.7. All our models are trained end-to-end to predict the full phonetic transcription at once, with the Autoseg criterion [Collobert et al., 2016], using stochastic gradient descent. We compare all models using either the baseline mel-filterbanks as input or our learnable Time-Domain filterbanks front-end. We perform the same grid-search for both mel-filterbanks baselines and models trained on Time-Domain filterbanks, using learning rates in $(0.0003, 0.003)$ for the model and learning rates in $(0.03, 0.003)$ for the Autoseg criterion, training every model for 2000 epochs. We use the standard dev set for early stopping and hyperparameter selection.

5.3.2 Different types of Time-Domain filterbanks

Throughout our experiments, we tried four different settings for the Time-Domain filterbanks layers:

- Fixed: Initialize the layers to match mel-filterbanks and keep their parameters fixed when training the model
- Learn-all: Initialize the layers and let the filterbank and the averaging be learned jointly with the model
- Learn-filterbank: Start from the initialization and only learn the filterbank with the model, keeping the averaging fixed to a squared Hann window
- Randinit: Initialize the layers randomly and learn them with the network

Model	Input	Dev PER	Test PER
Hybrid HMM/CNN + Maxout [Tóth, 2015]	mel + Δ + ΔΔ	13.3	16.5
Wavenet [Van Den Oord et al., 2016]	wav	-	18.8
CNN + CRF on raw speech [Palaz et al., 2013a]	wav	-	29.2
CNN-Conv2D-10L-Maxout [Zhang et al., 2017]	mel	16.7	18.2
Attention-based model [Chorowski et al., 2015]	mel + Δ + ΔΔ	15.8	17.6
LSTM + Segmental CRF [Lu et al., 2016]	mel + Δ + ΔΔ	-	18.9
LSTM + Segmental CRF [Lu et al., 2016]	MFCC + LDA/MLLT/MLLR	-	17.3
CNN-5L-ReLU-do0.5	mel	18.4	20.8
CNN-5L-ReLU-do0.5 + TD-filterbanks	wav	18.2	20.4
CNN-5L-ReLU-do0.7	mel	17.8	20.6
CNN-5L-ReLU-do0.7 + TD-filterbanks	wav	17.3	20.3
CNN-8L-PReLU-do0.7	mel	16.2	18.1
CNN-8L-PReLU-do0.7 + TD-filterbanks	wav	15.6	18.1
CNN-8L-PReLU-do0.7 + TD-filterbanks + preemp	wav	15.6	18.0

Table 5.3: PER (Phone Error Rate) on TIMIT, in percentages. "mel" stands for mel-filterbanks, while "TD-filterbanks" stands for Time-Domain filterbanks. All models but [Tóth, 2015] and [Van Den Oord et al., 2016] are trained in an end-to-end fashion.

Table 5.2 shows comparative performance of an identical architecture trained on the four types of Time-Domain filterbanks. We can observe that training on fixed layers moderately worsens the performance, we hypothesize that this is due to the absence of mean-variance normalization on top of Time-Domain filterbanks as is performed on mel-filterbanks (we will integrate this normalization to our learnable frontend in Chapter 6). A striking observation is that a model trained on Time-Domain filterbanks initialized randomly performs considerably worse than all other models. This shows the importance of the Gabor initialization. Finally, we observe better results when learning the filterbank only compared to learning the filterbank and the averaging but depending on the architecture it was not clear which one performs better. Moreover, when learning both complex filters and averaging, we observe that the learned averaging filters are almost identical to their initialization. Thus, in the following experiments, we choose to use the Learn-filterbank mode for the Time-Domain filterbanks.

5.3.3 Results

We report PER on the standard dev and test sets of TIMIT. For each architecture, we can observe that the model trained on Time-Domain filterbanks systematically outperforms the equivalent model trained on mel-filterbanks, even though we constrained our Time-Domain

filterbanks such that they are comparable to the mel-filterbanks (number of filters, window size, window stride). This shows that by only learning a new bank of 40 filters, we can outperform the mel-filterbanks for phone recognition. This gain in performance is obtained at a minimal cost in terms of number of parameters: even for the smallest architecture, the increase in number of parameters in switching from mel-filterbanks to Time-Domain filterbanks is 0.31%. We also compare to baselines from the literature. One baseline trained on the waveform gets a PER of 29.1% on the test set, which is in a range $8.8\% - 11.1\%$ absolute above our models trained on the waveform. The Wavenet architecture, also trained on the waveform, yields a PER of 18.8, which is higher than our best models despite the fact that it uses the phonetic alignment from an HMM, and an auxiliary prediction loss. Our best model on the waveform also outperforms a 2-dimensional CNN trained on mel-filterbanks and an LSTM trained on mel-filterbanks with derivatives.

5.4 Adding a learnable pre-emphasis layer

As described in Section 1.2.1, the first step in the computation of mel-filterbanks is typically the application of a pre-emphasis layer to the raw signal. Pre-emphasis is a convolution with a first-order high-pass filter of the form $y[n] = x[n] - \alpha x[n-1]$, with α typically equal to 0.97. This operation can be performed by a convolutional layer of kernel size 2 and stride 1, that can be plugged below time-domain filterbanks, initialized with weights $[-0.97 \quad 1]$, then learned with the network. By adding this learnable pre-emphasis layer below the Time-Domain filterbanks, and learning it jointly with the complex convolution, the lowpass filter, and the rest of the acoustic model we reach 18% PER on the test set.

5.5 Analysis of learnt filters

We analyze filters learned by the first layer of the CNN-8L-PReLU-do0.7 + Time-Domain filterbanks model. Examples of learned filters are shown in Figure 5-2. The magnitude of the frequency response for each of the 40 filters is plotted in Figure 5-3. Overall, the filters tend to be well localized in time and frequency, and a number of filters became asymmetric during the learning process, with a sharp attack and slow decay of the impulse response. This is a type of asymmetry also found in human and animal auditory filters estimated from

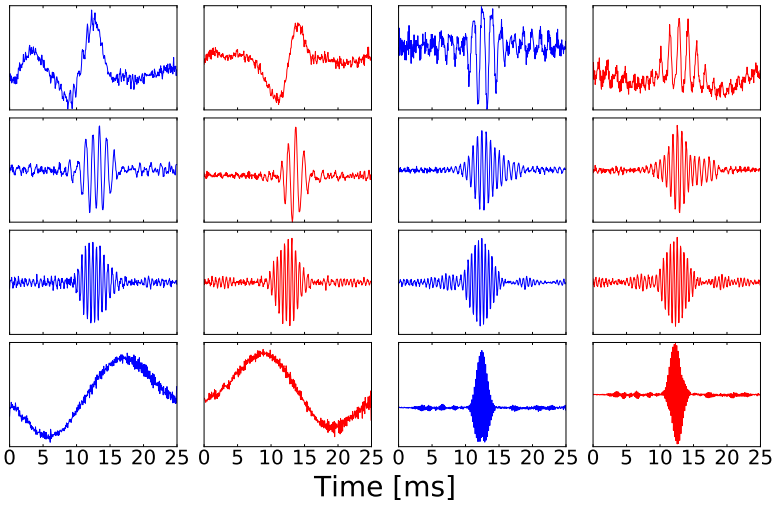


Figure 5-2: Examples of learnt filters. Filters' real parts in blue; imaginary part in red.

behavioral and physiological data [Smith and Lewicki, 2006]. In Figure 5-3, we further see that the initial mel-scale of frequency is mostly preserved, but that a lot of variability in the filter bandwidths is introduced.

A prominent question is whether the analyticity of the initial filterbank is preserved throughout the learning process even though nothing in our optimization method is biased towards keeping filters analytic. A positive answer would suggest that complex filters in their full generality are not necessary to obtain the increase in performance we observed. This would be especially interesting because, unlike arbitrary complex filters, analytic filters have a simple interpretation in terms of real-domain signal processing: taking the squared modulus of the convolution of a real signal with an analytic filter performs a sub-band Hilbert envelope extraction [Flanagan, 1980].

A signal is analytic if and only if it has no energy in the negative frequencies. Accordingly, we see in Figure 5-3 that there is zero energy in this region for the initialization filterbank. After learning, a moderate amount of energy appears in the negative frequency region for certain filters. To quantify this, we computed for each filter the ratio r_a between the energy in negative versus positive frequency components [1]. This ratio is 0 for a perfectly analytic filter and 1 for a purely real filter. We find an average r_a for all learned filters of .26.

[1] Our model cannot identify if a given filter plays the role of the real or imaginary part in the associated complex filter. We chose the assignment yielding the smallest r_a.

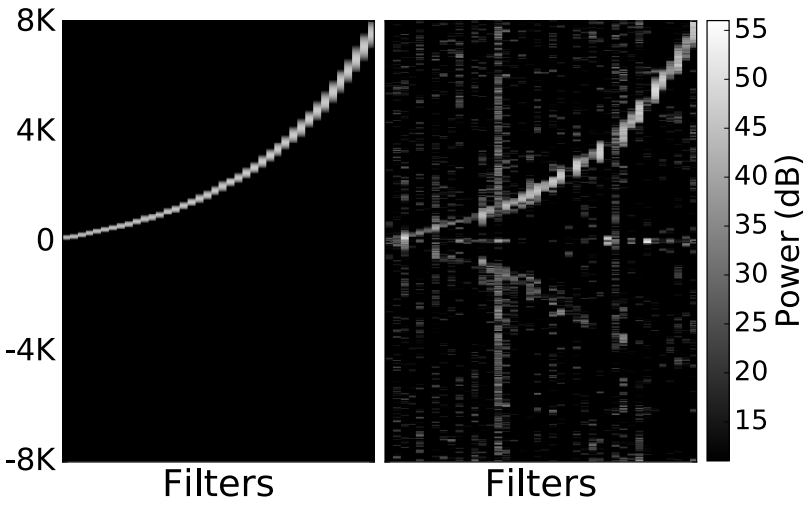


Figure 5-3: Heat-map of the magnitude of the frequency response for initialization filters (left) and learned filters (right).

Filters with significant energy in negative frequencies are mostly filters with an intermediate preferred frequency (between 1000Hz and 3000Hz) and their negative frequency spectrum appears to be essentially a down-scaled version of their positive frequency spectrum.

5.6 Conclusion

In this chapter, we introduced Time-Domain filterbanks, a lightweight architecture which, at initialization, approximates the computation of mel-filterbanks and can then be fine-tuned with an end-to-end phone recognition system. With a number of parameters comparable to standard mel-filterbanks, a Time-Domain filterbanks front-end is consistently better in our experiments. Learning all linear operations in the mel-filterbanks derivation, from pre-emphasis up-to averaging provides the best model. However, phonetic recognition on TIMIT could be labeled as a "toy" task, and we are yet to show the performance of Time-Domain filterbanks on a real speech recognition task. Moreover, we would like to benchmark our learnable frontend against other models from the literature. This is why in Chapter 6, we perform large scale experiments with Time-Domain filterbanks to test if a new state-of-the-art can be achieved by training from the waveform.

Chapter 6

End-to-End Speech Recognition from the Raw Waveform

This chapter is based on End-to-end Speech Recognition from the Raw Waveform [Zeghidour et al., 2018a], accepted for poster presentation at Interspeech 2018, and a joint work with Nicolas Usunier, Gabriel Synnaeve, Ronan Collobert and Emmanuel Dupoux.

6.1 Introduction

In the previous chapter, we showed promising results on a phone recognition task on TIMIT. However, TIMIT is a small dataset and its test set only contains 192 sequences. This dampens the significance of the differences in performance that we observe. Moreover, and as explained in Section 1.3, models trained on the raw waveform have not been proven to improve on speech features on large-scale, end-to-end speech recognition in clean recording conditions on English – admittedly one of the tasks for which mel-filterbanks have been the most extensively tuned. This is why we are interested in performing experiments at a larger scale.

Hoshen et al. [2015] and Sainath et al. [2015a] proposed a learnable frontend inspired by gammatone filterbanks [Schlüter et al., 2007], which achieved similar or better results than comparable mel-filterbanks on multichannel speech recognition and on far-field/noisy recording conditions. Noticing the analogies between the model proposed by Hoshen et al. [2015] and Sainath et al. [2015a] (that we will refer to as "gammatone-based frontend" to

avoid any confusion with fixed gammatone filterbanks), we present a systematic comparison of these two learnable frontends, and evaluate them against mel-filterbanks within an end-to-end training pipeline on the Wall Street Journal dataset. Our main contributions and results are the following:

1. A mean-variance normalization layer on top of the log nonlinearity of learnable filterbanks appears to be critical for the efficient learning of the gammatone-based architecture, and makes the training of Time-Domain filterbanks faster;

2. The low-pass filter previously used in the Time-Domain filterbanks stabilizes the training of the gammatone-based frontend, compared to the max-pooling that was originally proposed by Hoshen et al. [2015] and Sainath et al. [2015a];

3. For Time-Domain filterbanks, keeping the low-pass filter fixed during training allows to efficiently learn the filters from a random initialization, whereas the results of Chapter 5 with random initialization of both the filters and the low-pass filter showed poor performances compared to a suitable initialization;

4. With these modifications, both frontends improve against the mel-filterbanks baseline on Word Error Rate on the Wall Street Journal dataset, in similar conditions (same number of filters, same end-to-end training convolutional architecture). This is the first time learnable filterbanks improve against a strong mel-filterbanks baseline on a large vocabulary, speech recognition task under clean recording conditions.

The next section describes the learnable frontend architectures. Then, we present the end-to-end convolutional architecture used to perform the comparisons, and analyze the results of our comparative studies.

6.2 Time-Domain filterbanks and gammatone-based frontend

In this chapter we compare Time-Domain filterbanks to the gammatone-based frontend from Hoshen et al. [2015] and Sainath et al. [2015a]. They are described side-by-side in Table 6.1.

These two frontends can be used as a direct replacement for mel-filterbanks in any learning pipeline: they are convolutional architectures that take the raw waveform as input

	TD-FILTERBANKS	GAMMATONE-BASED
Conv *(#in-#out-width-stride)*	1-80-400-1	1-40-400-1
non-linearity	sq. L2-Pooling	ReLU
low-pass filter *(wdth=400, strd=160)*	sq. Hann	max-pooling or sq. Hann
log-compression	log(1+abs(.))	log(0.01+abs(.))
normalization	mean-var. per-channel per-sentence	

Table 6.1: Architectures of the two trainable filterbanks. Values of width and strides are given to match the standard settings of mel-filterbanks for waveform sampled at 16kHz. The convolution for the scattering-based architecture uses 80-real valued output channels and squared L2-pooling on the feature dimension to emulate a complex-valued convolution with 40 filters followed by a squared modulus operator. Thus, after the nonlinearity, both architectures have 40 filters. In Chapter 5, we use 1 to prevent log(0) and Hoshen et al. [2015] and Sainath et al. [2015a] use 0.01. We kept the values initially used by the authors of the respective contributions and did not try alternatives. We believe it has little impact on the final performance.

and output 40 channels every 10ms. In both cases, they can directly be compared with standard mel-filterbanks, simply by changing the features stage of a neural-network-based acoustic model. The filters are then nothing more than an additional layer to the neural network and are learnt by backpropagation with the rest of the acoustic model.

In both architectures, a convolutional layer with window length 25ms (to match the standard frame size used in mel-filterbanks) is applied with a stride of 1 sample, and is followed by a nonlinearity to give 40 output channels for each sample. Then, a pooling operator of width 25ms with a stride of 10ms performs low-pass filtering and decimation. Finally, a log non-linearity reproduces the dynamic range compression of mel-filterbanks. The parameters to be learnt are the convolution filters, and possibly the weights of the low-pass filters.

The two architectures differ by the choices of each layer of computation. Time-Domain filterbanks use 40 complex-valued filters with a square modulus operator as non-linearity. Low-pass filtering is then performed by multiplying each output channel by a squared Hann window so that, when using suitable Gabor wavelets as convolution filters, the architecture closely approximates mel-filterbanks computed on the power spectrum. Hoshen et al. and Sainath et al. use 40 real-valued filters with ReLU non-linearity, and rely on gammatones

as filter values to approximate mel-filterbanks [Hoshen et al., 2015, Sainath et al., 2015a]. In their work, they use a max-pooling operator for low-pass filtering.

The number of filters (40), the convolution and pooling width of 25ms, as well as the decimation of 10ms are not necessarily the optimal parameters of either trainable architecture, but these are the standard settings of mel-filterbanks (and likely the best settings for these features on standard speech recognition datasets). We keep these values fixed for the trainable architectures, so that the comparison to mel-filterbanks is carried out in the setting most favorable for the non-learnable baseline.

In the next subsections, we describe the improvements we propose for these architectures: the low-pass filter and the addition of instance normalization.

6.2.1 Low-pass filtering

Time-Domain filterbanks use a squared Hann window per channel as a low-pass filter, whereas the original papers describing the gammatone-based frontend used a max-pooling. To make sure the low-pass filter is not responsible for notable differences between the two approaches we experiment with the squared Hann window on both architectures. For both architectures, we also propose to keep this low-pass filter fixed when learning the convolution filter weights from a random initialization, a setting that was not explored in Chapter 5, in which we also randomly initialized the second convolutional layer of Time-Domain filterbanks.

6.2.2 Instance normalization

More importantly, we noticed that a per-channel per-sentence mean-variance normalization after log-compression is important for the baseline mel-filterbanks. Consequently, we propose to add a mean-variance normalization layer on both trainable architectures, performed for each of the 40 channels independently on each sentence. Coincidently, this corresponds to an instance normalization layer [Ulyanov et al., 2016], which has been shown to stabilize training in other deep learning contexts.

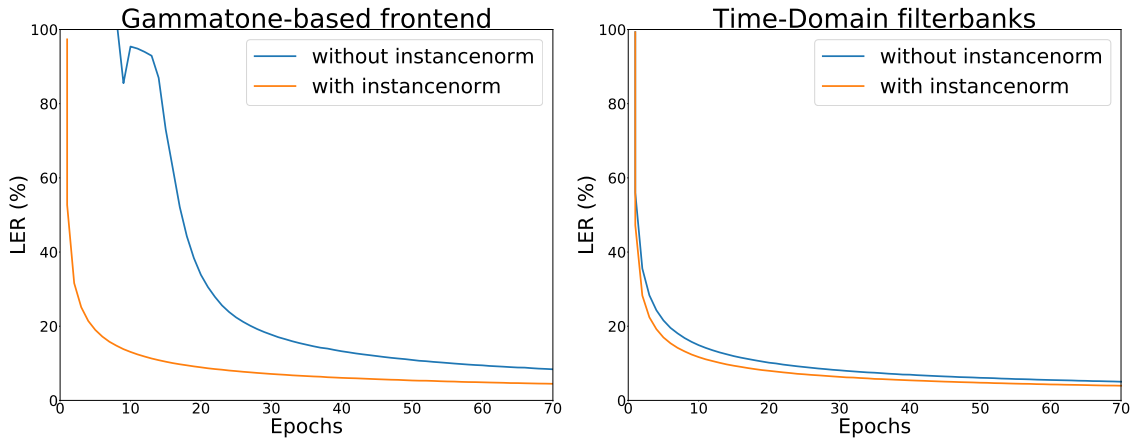


Figure 6-1: Training Letter Error Rate (LER) for the same acoustic model trained either on the gammatone-based frontend (left) or the Time-Domain filterbanks (right), with and without instance normalization.

6.3 Experimental setup

The experiments compare different versions of the learnable frontends against mel-filterbanks, on a single deep convolutional network architecture for the acoustic model. The experiments are carried out on the open vocabulary task of the Wall Street Journal dataset [Paul and Baker, 1992], using the subset si284 for training, nov93-dev for validation, and nov92-eval for testing. Training is performed end-to-end on letters. We evaluate in both Letter and Word Error Rates. All our experiments use the open source code of wav2letter [Collobert et al., 2016]. In the next subsections, we describe the model, the different variants we tested and the hyperparameters.

6.3.1 Acoustic model

Taking either mel-filterbanks or a learnable frontend as input, the acoustic model is a convolutional network with gated linear units (GLU) [Dauphin et al., 2017] trained to predict sequences of letters with the ASG loss function [Collobert et al., 2016], following Liptchinsky et al. [2017]. The model is a smaller version of the convolutional network used by Liptchinsky et al. [2017] since they train on the larger LibriSpeech dataset. Using the syntax C-input channels-output channels-width, the architecture we use has the structure C-40-200-13/C-100-200-3/C-100-200-4/C-100-250-5/C-125-250-6/C-125-300-7/C-150-350-8/C-175-400-9/

C-200-450-10/C-225-500-11/C-250-500-12/C-250-500-13/C-250-600-14/C-300-600-15/C-300-750-21/C-375-1000-1. All convolutions have stride 1. The number of input channels of the $n+1$th convolution is half the size of the output of the n-th convolution because of the GLU. There are GLU layers with a dropout [Hinton et al., 2012b] of 0.25 after each convolution layer. There is an additional linear layer to predict the final letter probabilities. When predicting letters, the training and decoding are performed as in [Liptchinsky et al., 2017]. When predicting words, we use a 4-gram language model trained on the standard LM data of WSJ [Paul and Baker, 1992] and perform beam search decoding, as in [Liptchinsky et al., 2017].

6.3.2 Variants

We compare the two architectures of learnable frontends along different axes: how to initialize the convolutions of the learnable frontend, the low-pass filter, and instance normalization.

Time-domain filterbanks

Initialization of the convolution weights random (rand), or Gabor filters (scatt) as described in Section 5.2.2;

Low-pass filter the squared Hann window (Hann-fixed), or a low-pass filter of same width and stride initialized with the weights of the squared Hann window but the weights are then learnt by backpropagation (Hann-learnt).

Gammatone-based frontend

Initialization of the convolution weights random (rand), or with gammatone filters (gamm) that match the impulse response of a reference open source implementation of gammatones [gam];

Low-pass filter max-pooling as in [Hoshen et al., 2015], or the squared Hann window (Hann-fixed).

6.3.3 Hyperparameters and training

For models trained on the raw waveform, the signal was first normalized with mean-variance normalization by sequence. The network is trained with stochastic gradient descent and weight normalization [Salimans and Kingma, 2016] for all convolutional layers except the front-ends. First, 80 epochs are performed with a learning rate of 1.4, then training is resumed for 80 additional epochs with a learning rate of 0.1. These hyperparameters were chosen from preliminary experiments as they seemed to work well for all architectures. Additional hyperparameters are the momentum and the learning rate for the training criterion, respectively chosen in $\{0, 0.9\}$ and $\{0.001, 0.0001\}$ [Collobert et al., 2016, Liptchinsky et al., 2017].

For Letter Error Rate (LER) evaluations, the hyperparameters are selected using the LER on the validation set, validating every epoch. For Word Error Rate (WER) evaluations, the hyperparameters are chosen on the validation set using the WER, validating every 10 epochs. The model selected on LER is also included for validation. The additional hyperparameters are the weight of the language model and the weight of word insertion penalty (see [Liptchinsky et al., 2017] for details). We set them between 5 and 8 by steps of 0.5, and between -2 and 0.5 by steps of 0.1, respectively. For hyperparameter selection, the beam size of the decoder is set to $2,500$; the final performances are computed with the selected hyperparameters but using a beam size of $25,000$.

6.4 Experiments

6.4.1 Baseline results

Table 8.2 contains our results together with end-to-end baselines from the literature. [Chan and Lane, 2015] is the current state-of-the-art on this dataset. It is an HMM-based system which uses a combination of convolutional, recurrent and fully connected layers, as well as speaker adaptation, and reaches 3.5% WER on nov92-eval. [Amodei et al., 2015] is state-of-the-art among end-to-end models (trained without alignment); it is given as a topline but uses much more training data ($\sim 12,000h$ of speech) so the results are not comparable. [Chorowski and Jaitly, 2016, Graves and Jaitly, 2014, Kim et al., 2017, Miao et al., 2015] are representative results in terms of WER and LER from the literature of end-to-end models

MODEL			NOV93-DEV		NOV92-EVAL	
			LER	WER	LER	WER
SOTA – SPEECH FEATURES						
CNN-DNN-BLSTM-HMM [Chan and Lane, 2015]			-	6.6	-	3.5
Deep Speech 2 [Amodei et al., 2015]			–	–	–	3.6
– (+ additional data)						
RNN-WER - tri. LM [Graves and Jaitly, 2014]			–	–	–	8.2
RNN - WSFT decoding [Miao et al., 2015]			–	–	–	7.3
Seq2Seq + tri. LM [Chorowski and Jaitly, 2016]			–	9.7	–	6.7
Multi-task CTC/att [Kim et al., 2017]			11.3	–	7.3	–
Att + RL [Tjandra et al., 2017b]			–	–	6.1	
SOTA – WAVEFORM						
Seq2Seq + mel pretraining [Tjandra et al., 2017a]			–	–	6.5	–
gamm (learnt)/gamm/max-pool			8.9	12.9	6.4	8.8
– (without inst. norm.)						
FRONTEND	FILTER INIT	LOWPASS	NOV93-DEV		NOV92-EVAL	
			LER	WER	LER	WER
mel-filterbanks			6.9	9.5	4.9	6.6
gamm (learnt)	gamm	Hann-fixed	6.9	9.1	4.9	5.9
		max-pool	7.2	9.3	4.9	6.0
	rand	Hann-fixed	7	8.9	4.9	5.9
		max-pool	7.2	9.2	5.1	6.3
scatt (learnt)	scatt	Hann-fixed	6.7	8.3	4.6	6.1
		Hann-learnt	6.7	8.9	4.5	6.3
	rand	Hann-fixed	6.8	8.5	4.7	5.7
		Hann-learnt	6.9	8.9	4.9	5.8

Table 6.2: Results (%) on the open vocabulary task of the WSJ dataset. *(i)* SOTA – speech features: for state-of-the-art and representative baselines using speech features (mel-filterbanks, spectrograms or MFCC), *(ii)* SOTA-waveform: state-of-the-art from the raw waveform, including our own implementation of vanilla gammatone-based frontend without instance normalization, and *(iii)* our baseline and the different variants of the learnable frontends (with instance normalization) studied in this chapter.

trained on speech features from 2014-2017, in chronological order. [Tjandra et al., 2017b] and [Tjandra et al., 2017a] were the current state-of-the-art in LER on speech features and from the waveform respectively, at time of publication of [Zeghidour et al., 2018a]. These comparisons validate our baseline model trained on mel-filterbanks as a strong baseline in light of recent results, as it outperforms the state-of-the-art in LER by a significant margin (4.9% for our best model vs 6.1% for [Tjandra et al., 2017b]), and achieves a test WER of 6.6%, better than all other end-to-end baselines ([Zhou et al., 2018] and [Ghahremani et al., 2016] report WER that are below our 6.6% but are on easier closed vocabulary tasks).

6.4.2 Instance normalization

As described in Section 6.2.2, we evaluate the integration of instance normalization after the log-compression into the learnable frontends which was not used in previous work [Ghahremani et al., 2016, Hoshen et al., 2015, Sainath et al., 2015a] not in our TIMIT experiments in Chapter 5, but is used in our mel-filterbanks baseline. Figure 6-1 shows training LER as a function of the number of epochs for scattering-based and gammatone-based filterbanks models, with and without instance normalization. We can see that this normalization drastically improves the training stability of the gammatone-based model, while it moderately improves the Time-Domain filterbanks model. We observed a positive impact of instance normalization in all settings, and so only report as a reference the results of our implementation of a vanilla gammatone-based trainable filterbanks followingHoshen et al. [2015] and Sainath et al. [2015a]. Comparing gammatone (learnt)/gamm/max-pool without instance norm (under SOTA – waveform) to the results of gammatone (learnt)/gamm/max-pool in Table 8.2, we see a significant improvements of both LER and WER due to instance normalization, with an absolute reduction in LER and WER of 1.5% and 2.8% respectively.

6.4.3 Impact of the low-pass filter

For low-pass filtering, we first compare the Hann-fixed setting to max-pooling for gammatone-based filterbanks (as max-pooling was previously used by Hoshen et al. [2015] and Sainath et al. [2015a]), and to Hann-learnt for Time-Domain filterbanks, all with instance normalization. The tendency is that the Hann-fixed setting consistently improves the results in LER and WER of both learnable frontends. More importantly, using either an Hann-fixed or Hann-learnt filter when learning Time-Domain filterbanks from a random initialization removes the gap in performance with the Gabor wavelet initialization that was observed in Chapter 5, where the lowpass filter was also initialized randomly. This is an important result since carefully initializing the convolutional filters is both technically non-trivial, and also relies on the prior knowledge of mel-filterbanks. We believe the ability to use random initialization is an important first step for more extensive tuning of learnable frontends (e.g., trying different numbers of filters, decimation or convolution width).

Compared to the literature, replacing the max-pooling by a low-pass filter and adding an instance normalization layer leads to a 23% relative improvement in LER and a 33%

MODEL	PRE-EMP	NOV93-DEV		NOV92-EVAL	
		LER	WER	LER	WER
gamm	no pre-emp	6.9	9.1	4.9	5.9
(learnt)	pre-emp	6.8	9	4.7	5.7
scatt	no pre-emp	6.7	8.3	4.6	6.1
(learnt)	pre-emp	6.5	8.7	4.5	5.7

Table 6.3: Comparison of models trained with or without a learnable pre-emphasis layer. All models are initialized either with the scattering or gammatone initialization, and the pooling function is a fixed squared Hann window.

relative improvement in WER on nov92-eval on the gammatone-based fronted, a significant improvement compared to the existing approach of Hoshen et al. [2015] and Sainath et al. [2015a]. Our models trained on the waveform also exhibit a gain in performance in LER of $22-31\%$ relative compared to the state-of-the-art end-to-end model trained on the waveform with its first 9 layers being pre-trained for mel-filterbanks reconstruction [Tjandra et al., 2017a], and outperform various end-to-end models trained on speech features, both in LER [Kim et al., 2017, Tjandra et al., 2017b] and WER [Chorowski and Jaitly, 2016, Graves and Jaitly, 2014, Miao et al., 2015].

6.4.4 Learnable frontends vs mel-filterbanks

Comparing both learnable frontends with instance normalization to the mel-filterbanks baseline, we observe that the performances of the Hann-fixed settings and of the mel-filterbanks are comparable in terms of LER. However, we observe a consistent improvement in terms of WER of all learnable frontends. To the best of our knowledge, this is the first time a significant improvement in terms of WER relatively to comparable mel-filterbanks has been shown on a large vocabulary task under clean recording conditions. Some improvements on the clean test of the Switchboard dataset have previously been observed by Ghahremani et al. [2016], but their comparison point is MFCC rather than mel-filterbanks and the number of filters of the trainable architecture differs from their MFCC baseline.

6.4.5 Adding a learnable pre-emphasis layer

As in Section 5.4, we also experiment with a learnable pre-emphasis layer integrated into the Time-Domain filterbanks. In Table 6.3, we compare the performance of identical models (all using a fixed Hann window, and a gammatone or Gabor initialization) with and without pre-emphasis. We observe a gain on both LER and WER (except on nov93-dev WER/scatt) when learning a pre-emphasis with the rest of the frontend.

6.5 Conclusion

In this chapter, we presented a systematic comparison of Time-Domain filterbanks and a previously proposed gammatone-based frontend, which clarifies good practices and identifies better architectures to learn from raw speech. Our results show that adding an instance normalization layer on top of the learnable frontend is critical for learning gammatone-based architectures, and speeds up learning of acoustic models trained on Time-Domain filterbanks. Second, the use of a fixed squared Hann window as low-pass filter is critical to train Time-Domain filterbanks from a random initialization of the filters, and improves on max-pooling for the gammatone-based frontend. With these two improvements, we observe a consistent reduction of WER against comparable mel-filterbanks on the open vocabulary task of the WSJ dataset, in the setting of speech recognition under clean recording condition – most likely the setting on which mel-filterbanks have been the most heavily tuned. In the next Chapter, we show that we can substantially improve furthermore by combining our current approach with a convolutional language model.

Chapter 7

Fully Convolutional Speech Recognition

This chapter is based on the material of Fully Convolutional Speech Recognition [Zeghidour et al., 2018b], an arxiv preprint and a joint work with Qiantong Xu (equal contribution), Vitaliy Liptchinsky, Nicolas Usunier, Gabriel Synnaeve and Ronan Collobert.

7.1 Introduction

Recent work on convolutional neural network architectures showed that they are competitive with recurrent architectures even on tasks where modeling long-range dependencies is critical, such as language modeling [Dauphin et al., 2017], machine translation [Gehring et al., 2017a,b] and speech synthesis [Van Den Oord et al., 2016]. In end-to-end speech recognition however, recurrent architectures are still prevalent for acoustic and/or language modeling [Amodei et al., 2015, Chan et al., 2015, Graves and Jaitly, 2014, Mikolov et al., 2010, Zeyer et al., 2018].

There is a history of using convolutional networks in speech recognition, but only as part of an otherwise more traditional pipeline. They have been first introduced as TDNNs to predict phoneme classes [Waibel et al., 1989], and later to generate HMM posteriorgrams [Abdel-Hamid et al., 2014]. They have more recently been used in end-to-end frameworks, but only in combination with recurrent layers [Amodei et al., 2015], or n-gram language models [Liptchinsky et al., 2017] (see Chapter 6), or for phone recognition [Zhang et al.,

2017] (see Chapter 5). Nonetheless, as we showed in Section 1.3 and Chapter 6, convolutional architectures are prevalent when learning from the raw waveform, because they naturally model the computation of standard features such as mel-filterbanks. Given the evidence that they are also suitable on long-range dependency tasks, we expect convolutional neural networks to be competitive at all levels of the speech recognition pipeline.

In this chapter, we present a fully convolutional approach to end-to-end speech recognition. Building on Time-Domain filterbanks and convolutional acoustic models described in Chapter 6, and convolutional language models [Dauphin et al., 2017], this chapter has four main contributions:

- We present the first application of convolutional language models to speech recognition. They yield significant improvements over 4-gram language models on both Wall Street Journal (WSJ) and Librispeech datasets.

- We show that fully convolutional approaches are competitive with approaches based on recurrent neural networks. In particular, on Librispeech, we improve by more than 2% absolute Word Error Rate the results of DeepSpeech 2 [Amodei et al., 2015] and of the best sequence-to-sequence model [Zeyer et al., 2018].

- We present the first state-of-the-art results (among end-to-end systems) on a large, publicly available dataset (Librispeech) that use end-to-end learning from the raw waveform. On WSJ, we significantly improve over the best previous results presented in Chapter 6 and match the current state-of-the-art, a DNN-HMM system.

- On Librispeech, learning the frontend has a larger impact in noisy than in clean recording conditions. These results corroborate previous observations on the VoiceSearch dataset [Hoshen et al., 2015, Sainath et al., 2015a], and give additional evidence that mel-filterbanks are suboptimal in the noisy setting.

7.2 Model

Our approach, described in this section, is illustrated in Fig. 7-1.

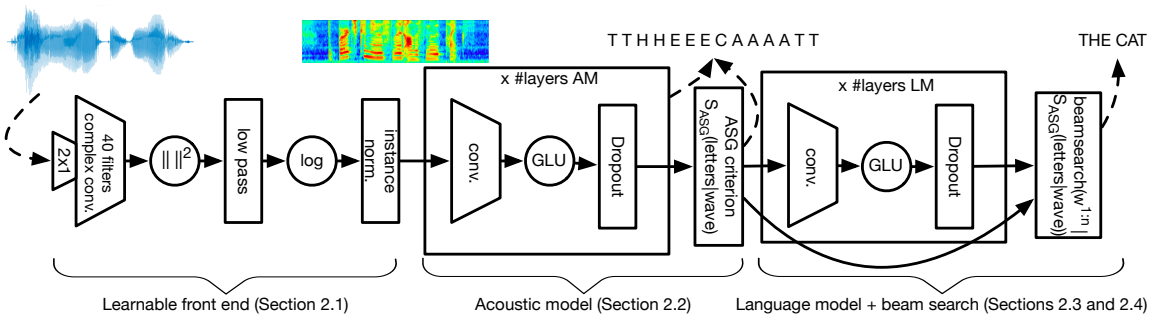


Figure 7-1: Overview of the fully convolutional architecture.

7.2.1 Convolutional Frontend: Time-Domain filterbanks

Following the systematic comparison made in Chapter 6, we consider the best architecture, Time-Domain filterbanks with learnable pre-emphasis. The learnable frontend contains first a convolution of width 2 that emulates the pre-emphasis step used in mel-filterbanks. It is followed by a complex convolution of width 25 ms and k filters. After taking the squared absolute value, a low-pass filter of width 25 ms and stride 10 ms performs decimation. The frontend finally applies a log-compression and a per-channel mean-variance normalization (equivalent to an instance normalization layer [Ulyanov et al., 2016]). The low-pass filter is kept constant to a squared Hann window, and the complex convolutional layer is initialized randomly. In addition to the $k = 40$ filters used in Chapter 6, we also experiment with $k = 80$ filters. Notice that since the stride is the same as for mel-filterbanks, acoustic models on top of the learnable frontends can also be applied to mel-filterbanks, simply modifying the number of input channels if $k \neq 40$.

7.2.2 Convolutional Acoustic Model

The acoustic model is a convolutional neural network with gated linear units [Dauphin et al., 2017], which is fed with the output of the learnable frontend. Following [Liptchinsky et al., 2017] and Chapter 7, the networks use a growing number of channels, and dropout [Hinton et al., 2012b] for regularization. These acoustic models are trained to predict letters directly with the Auto Segmentation Criterion (ASG) [Collobert et al., 2016]. The only differences between the WSJ and Librispeech models are their depth, the number of feature maps per layer, the receptive field and the amount of dropout.

7.2.3 Convolutional Language Model

The convolutional language model (LM) is the GCNN-14B from [Dauphin et al., 2017], which achieved competitive results on several language modeling benchmarks, and operates at the word level. The network contains 14 convolutional residual blocks [He et al., 2016] with a growing number of channels, and uses gated linear units as activation function.

The language model is used to score candidate transcriptions in addition to the acoustic model in the beam search decoder described in the next section. Compared to n-gram LMs, convolutional LMs allow for much larger context sizes. Our detailed experiments in Setion 7.4.2 study the effect of context size on the final speech recognition performance.

7.2.4 Beam-search decoder

We use the beam-search decoder presented in [Liptchinsky et al., 2017] to generate word sequences given the output from our acoustic model. The decoder finds the word transcription W to maximize:

$$AM(W) + \alpha \; log \; P_{lm}(W) + \beta|W| - \gamma|\{i|\pi_i = \langle sil \rangle\}|,$$

where π_i is the value for the ith frame in the path leading to W and $AM(W)$ is the (unnormalized) acoustic model score of the transcription W. The hyperparameters $\alpha, \beta, \gamma \geq 0$ respectively control the weight of the language model, the word insertion reward, and the silence insertion penalty. Other parameters are the beam size and the beam score, a threshold under which candidates are discarded even if the beam is not full. These are chosen according to a trade-off between (near-)optimality of the search and computational cost.

7.3 Experiments

We evaluate our approach on the large vocabulary task of the Wall Street Journal (WSJ) dataset [Paul and Baker, 1992], which contains 80 hours of clean read speech, and Librispeech [Panayotov et al., 2015], which contains 1000 hours with separate train/dev/test splits for clean and noisy speech. Each dataset comes with official textual data to train language models, which contain 37 million tokens for WSJ, 800 million tokens for Librispeech. Our language models are trained separately for each dataset on the official text data only. These

datasets were chosen to study the impact of the different components of our system at different scales of training data and in different recording conditions.

The models are evaluated in Word Error Rate (WER). Our experiments use the open source codes of wav2letter[1] for the acoustic model, and fairseq[2] for the language model. More details on the experimental setup are given below.

Baseline Our baseline for each dataset follows [Liptchinsky et al., 2017]. It uses the same convolutional acoustic model as our approach but a mel-filterbanks frontend and a 4-gram language model.

Training/test splits On WSJ, models are trained on *si284*. *nov93dev* is used for validation and *nov92* for test. On Librispeech, we train on the concatenation of *train-clean* and *train-other*. The validation set is *dev-clean* when testing on *test-clean*, and *dev-other* when testing on *test-other*.

Acoustic model architecture The architecture for the convolutional acoustic model is the "high dropout" model from Liptchinsky et al. [2017] for Librispeech, which has 19 layers in addition to the frontend (mel-filterbanks for the baseline, or the learnable frontend for our approach). On WSJ, we use the lighter version used in Chapter 6, which has 17 layers. Dropout is applied at each layer after the frontend, following Chapter 6. The learnable frontend uses 40 or 80 filters.

Language model architecture As described in Section 7.2.3, we use the GCNN-14B model of Dauphin et al. [2017] with dropout at each convolutional and linear layer on both WSJ and Librispeech. We keep all the words (162K) in WSJ training corpus. For Librispeech, we only use the most frequent 200K tokens (out of 900K).

Hyperparameter tuning The acoustic models are trained following the experimental setup of Chapter 6, using SGD with a decreasing learning rate, weight normalization and gradient clipping at 0.2 and a momentum of 0.9. The language models are trained with Nesterov accelerated gradient [Sutskever et al., 2013]. Following [Dauphin et al., 2017], we also use weight normalization and gradient clipping.

The parameters of the beam search (see Section 7.2.4) α, β and γ are tuned on the validation set with a beam size of 2500 and a beam score of 26 for efficiency. Once α, β, γ are chosen, the test WER is computed with a beam size of 3000 and a beam score of 50.

[1] https://github.com/facebookresearch/wav2letter
[2] https://github.com/facebookresearch/fairseq

Model		nov93	nov92
E2E Lattice-free MMI [Hadian et al., 2018] *(data augmentation)*		-	4.1
CNN-DNN-BLSTM-HMM [Chan and Lane, 2015] *(speaker adaptation, 3k acoustic states)*		6.6	3.5
DeepSpeech 2 [Amodei et al., 2015] *(12k training hours AM, common crawl LM)*		5	3.6
Frontend	LM	nov93	nov92
mel-filterbanks	4-gram	9.5	6.6
mel-filterbanks	ConvLM	7.5	4.1
Time-Domain filterbanks (40 filters)	ConvLM	7.1	4.0
Time-Domain filterbanks (80 filters)	ConvLM	6.8	3.5

Table 7.1: WER (%) on the open vocabulary task of WSJ.

7.4 Results

7.4.1 Word Error Rate results

Wall Street Journal dataset

Table 7.1 shows Word Error Rates (WER) on WSJ for the current state-of-the-art and our models. The current best model trained on this dataset is an HMM-based system which uses a combination of convolutional, recurrent and fully connected layers, as well as speaker adaptation, and reaches 3.5% WER on *nov92*. DeepSpeech 2 shows a WER of 3.6% but uses 150 times more training data for the acoustic model and huge text datasets for LM training. Finally, the state-of-the-art among end-to-end systems trained only on WSJ, and hence the most comparable to our system, at time of publication of [Zeghidour et al., 2018b], uses lattice-free MMI on augmented data (with speed perturbation) and gets 4.1% WER. Our baseline system, trained on mel-filterbanks, and decoded with a n-gram language model has a 5.6% WER. Replacing the n-gram LM by a convolutional one reduces the WER to 4.1% , and puts our model on par with the current best end-to-end system. Replacing the speech features by Time-Domain filterbanks finally reduces the WER to 4.0% and then to 3.5% when doubling the number of learnable filters, improving over DeepSpeech 2 and matching the performance of the best DNN-HMM system.

Model		dev-clean	dev-other	test-clean	test-other
CAPIO (Single) [Han et al., 2017] *(speaker adapt., pronunciation lexicon)*		3.02	8.28	3.56	8.58
CAPIO (Ensemble) [Han et al., 2017] *(Combination of 8 systems)*		2.68	7.56	3.19	7.64
DeepSpeech 2 [Amodei et al., 2015] *(12k training hours AM, common crawl LM)*		-	-	5.83	12.69
Sequence-to-sequence [Zeyer et al., 2018]		3.54	11.52	3.82	12.76
Frontend	LM	dev-clean	dev-other	test-clean	test-other
mel-filterbanks	4-gram	4.26	13.80	4.82	14.54
mel-filterbanks	ConvLM	3.13	10.61	3.45	11.92
Time-Domain filterbanks (40 filters)	ConvLM	3.16	10.05	3.44	11.24
Time-Domain filterbanks (80 filters)	ConvLM	3.08	9.94	3.26	10.47

Table 7.2: WER (%) on Librispeech.

Librispeech dataset

Table 7.2 reports WER on the Librispeech dataset. The CAPIO [Han et al., 2017] ensemble model combines the lattices from 8 individual DNN-HMM systems (using both convolutional and LSTM layers), and is the current state-of-the-art on Librispeech. CAPIO (single) is the best individual system, selected either on dev-clean or dev-other. The sequence-to-sequence baseline is an encoder-decoder with attention and a BPE-level [Sennrich et al., 2015] LM, and currently the best end-to-end system on this dataset. We can observe that our fully convolutional model improves over CAPIO (Single) on the clean part, and is the current best end-to-end system on test-other with an improvement of 2.3% absolute (18% relative). Our system also outperforms DeepSpeech 2 on both test sets by a significant margin. An interesting observation is the impact of each convolutional block. While replacing the 4-gram LM by a convolutional LM improves similarly on the clean and noisier parts, learning the speech frontend gives similar performance on the clean part but significantly improves the performance on noisier, harder utterances, a finding that is consistent with previous literature [Hoshen et al., 2015]. Again, when doubling the number of Time-Domain filterbanks, we furthermore widen the gap with the mel-filterbanks baseline and the previous state-of-the-art, both on test-clean and test-other.

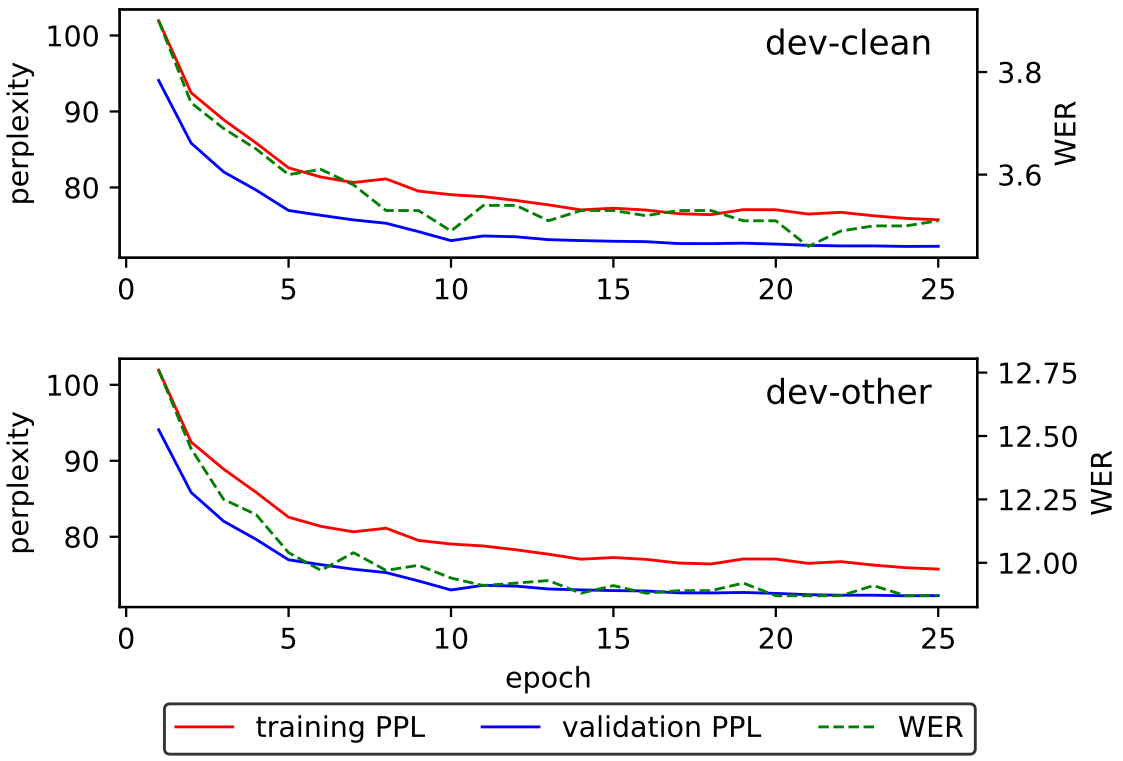


Figure 7-2: Evolution of WER (%) on Librispeech with the perplexity of the language model.

7.4.2 Analysis of the convolutional language model

Since our model uses convolutional language models for speech recognition systems for the first time, we present additional studies of the language model in isolation. These experiments use our best language model on Librispeech, and evaluations in WER are carried out using the baseline system trained on mel-filterbanks. The decoder parameters are tuned using the grid search described in Section 7.3, a beam size is fixed to 2500 and a beam score to 30.

Correlation between perplexity and WER Figure 7-2 shows the correlation between perplexity and WER as the training progresses. As perplexity decreases, the WER on both *dev-clean* and *dev-other* also decreases following the same trend. It illustrates that perplexity on the linguistic data is a good surrogate of the final performance of the speech recognition pipeline, as previously shown by Klakow and Peters [2002]. Architectural choices or hyperparameter tuning can thus be carried out mostly using perplexity alone.

Model Type	Context Length	WER	
		dev-clean	dev-other
4-gram	3	4.26	13.80
ConvLM	3	4.11	13.17
ConvLM	9	3.34	11.29
ConvLM	19	3.27	11.06
ConvLM	29	3.25	11.09
ConvLM	39	3.24	11.07
ConvLM	49	3.24	11.08

Table 7.3: Evolution of WER (%) on Librispeech with the context size of the language model.

Influence of context size By limiting the context passed into the LM from the decoder, Table 7.3 reports WER obtained for context sizes ranging from 3 (comparable to the n-gram baseline) to 50 for our best language model. The WER decreases monotonically until a context size of about 20, and then almost stays still. We observe that the convolutional LM already improves on the n-gram model even with the same context size. Increasing the context gives a significant boost in performance, with the major gains obtained between a context of 3 to 9 (-1.9% absolute WER).

7.5 Conclusion

We introduced the first fully convolutional pipeline for speech recognition, that can directly process the raw waveform and shows state-of-the art performance on Wall Street Journal and on Librispeech among end-to-end systems. This first attempt at exploiting convolutional language models in speech recognition shows significant improvement over a 4-gram language model on both datasets. Replacing mel-filterbanks by Time-Domain filterbanks gives additional gains in performance, that appear to be more prevalent on noisy data. Finally, we see a significant improvement from doubling the number of learnable filters.

Chapter 8

Learning to Detect Dysarthria from Raw Speech

This chapter is based on Learning to detect dysarthria from raw speech [Millet and Zeghidour, 2018], accepted for oral presentation at ICASSP 2019, and a joint work with Juliette Millet during her master internship at ENS under my supervision.

8.1 Introduction

In Chapters 5, 6, 7, we started from small scale systematic studies that compared mel-filterbanks and Time-Domain filterbanks, to the first state-of-the-art speech recognition system on Wall Street Journal and Librispeech that does not use speech features. The observation that Time-Domain filterbanks consistently outperform mel-filterbanks in a task for which they have been tuned, speech recognition, motivates extending their use to other tasks that use hardcoded features. In Section 1.4, we explained how paralinguistic classification systems, which recognize other characteristics from speech than its linguistic content, also use fixed, handcrafted features, such as mel-filterbanks or MFCCs, and/or low-level informations [Eyben et al., 2010], such as zero-crossing rate or harmonics-to-noise ratio (the number of descriptors can be as high as 6000 for a small speech segment [Schuller et al., 2017]). Training a classifier from these fixed coefficients requires performing a feature selection step, which has the limitation that it cannot retrieve useful information that would have been lost in the feature computation. Moreover, and as explained in Section 1.4, mel-

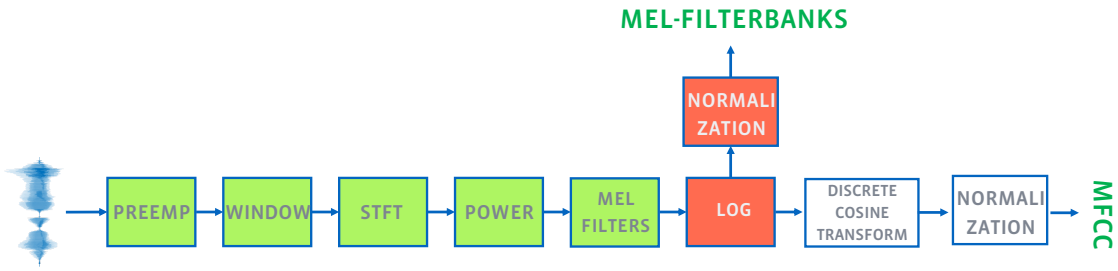


Figure 8-1: Computational steps that produce mel-filterbanks and MFCC features. In Green are the operations that are learnable (or to which an equivalent is learnable) in Time-Domain filterbanks. Red boxes are the operations that remain fixed during training.

filterbanks or MFCCs have been designed to mimic characteristics of human perception of speech. This is justified for speech recognition, since the human auditory system is robust to many sources of variability in the speech signal (constant or impulsive noise, speaking style and accent, room reverberation, etc.) [Morgan et al., 2004] that are challenging for automatic speech recognition. However, this observation also questions the appropriateness of using such features in tasks for which there is no proven performance of the human auditory system, and for which we could expect a system trained solely for this task to perform better. Among such tasks, there is a growing interest in automatically extracting information from speech for health care [Johnson et al., 2014, Little et al., 2009, Schuller et al., 2013b]. In this chapter, we propose to address one of these tasks: the detection of dysarthria from speech recordings. Rather than adopting a feature-driven approach that would require testing various combinations of fixed features, or training a model that would suffer from the biases of mel-filterbanks, we implement a system that can directly process raw speech and learn relevant features jointly with the dysarthria classifier, such that they are optimal for the task.

The TORGO database [Rudzicz et al., 2012] is a collection of annotated speech recordings and articulatory measurements from speakers with cerebral palsy (CP) or amyotrophic lateral sclerosis (ALS), as well as control patients. These pathologies are the cause of a motor speech disorder called dysarthria, which impedes the articulation and reduces the intelligibility of the speech that is produced. Mengistu and Rudzicz [2011a,b] and Kim et al. [2013] have used this database to provide speech recognition systems with robustness to dysarthria. [Kim et al., 2015] trains various linear classifiers on TORGO and the NKI CCRT corpus

[Clapham et al., 2012] to detect dysarthria. More recently, Bhat et al. [2017] have trained fully connected neural networks to classify the severity of the disease, using TORGO and the UASPEECH [Kim et al., 2008] database. All these models are trained on standard low-level features [Schuller et al., 2013b]. In this work we show that dysarthria detection benefits significantly from learning directly from the raw waveform with Time-Domain filterbanks.

Figure 8-1 shows the mel-filterbanks computation pipeline presented in Part I, and those of its components that we have made learnable so far. Time-Domain filterbanks allow replacing the pre-emphasis, windowing, and frequency filtering of mel-filterbanks by learnable components that all operate in the time-domain. The output of these operations is then passed through a log compression, and a mean-variance normalization (using an instance normalization [Ulyanov et al., 2016]), that remain fixed during training. However, as explained in Sections 1.2.4, various compression functions including logarithm, cubic root, or $10th$ root have been previously showed to perform better depending on the task (see Table 2 of [Schlüter et al., 2007]). In Section 1.2.6, we described how the choice of the type of normalization to apply to input features is also not trivial, and is task dependent. As we observed improvements in previous chapters by replacing the presumably biased linear operations of mel-filterbanks by learnable convolutional layer, we could thus also expect a gain from additionally learning a compression function and a normalization.

Wang et al. [2017] introduce a computational block, the Per Channel Energy Normalization (PCEN) that can learn a compression and a normalization factor per channel, and can be integrated into a neural network on top of speech features. It has been used in production ASR systems on fixed spectrograms [Battenberg et al., 2017], or bird vocalization and audio event classification from mel-filterbanks [Lostanlen et al., 2019]. This chapter presents a first tentative at learning jointly the PCEN with a learnable frontend.

In this chapter, we start from an attention-based model on mel-filterbanks, which already outperforms an equivalent model trained on low-level descriptors (LLDs). Our experiments show that by training a PCEN block on top of mel-filterbanks or replacing them by Time-Domain filterbanks, we get a gain in accuracy around 10% in absolute when training an identical neural network for dysarthria detection. Finally, by combining Time-Domain filterbanks and PCEN we propose the first fully learnable audio frontend, that can learn features, compression and normalization jointly with a neural network using backpropagation.

8.2 Model

8.2.1 Time-Domain filterbanks

As the first step of our computational pipeline, we use Time-Domain filterbanks (see 5 for a detailed description), As in Chapter 7, we choose the best performing system from Chapter 6, Time-Domain filterbanks with learnable pre-emphasis. However, the mel-filterbanks that we use for this task have different number of filters than for speech recognition. The learnable front-end contains first a learnable pre-emphasis layer, followed by a complex convolution of width 25ms and 64 filters. After taking the squared absolute value, a low-pass filter of width 25ms and stride 10ms performs decimation. The second convolution layer is kept fixed as a squared Hann window to perform lowpass filtering, and we use the Gabor initialization described in Section 5.2.2, due to the small size of the dataset. When used in combination with PCEN, we remove the log compression and instance normalization from the Time-Domain filterbanks architecture, as these operations will be learned by PCEN.

8.2.2 Per Channel Energy Normalization

Per Channel Energy Normalization (PCEN) is a learnable component introduced by Wang et al. [2017] which computes parametrized normalization and compression. It replaces the log-compression and the mean-variance normalization. With $E(t, f)$ the value of the feature f at time t, the computation of PCEN is:

$$PCEN(t, f) = (\frac{E(t, f)}{(\epsilon + M(t, f))^{\alpha}} + \delta)^r - \delta^r.$$

$M(t, f)$ is an exponential moving average of $E(\cdot, f)$ along the time axis, defined as:

$$M(t, f) = (1 - s)M(t - 1, f) + sE(t, f).$$

In our experiments, E is either the mel-filterbanks, or the output of Time-Domain filterbanks. α controls the strength of the normalization, the exponent r (typically in $[0, 1]$) defines the slope of the compression curve, s sets the spread of the moving average, and ϵ is a small scalar used to avoid division by zero. By backpropagation, we learn α, r, and δ with the rest of the model to learn a compression and normalization that fit the task at hand.

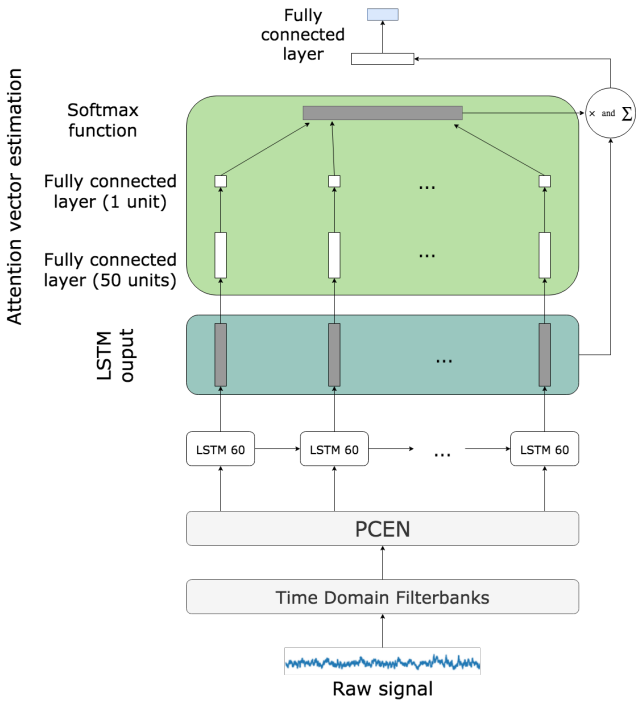


Figure 8-2: Proposed pipeline that learns jointly the feature extraction, the compression, the normalization and the classifier.

8.2.3 LSTM and Attention model

The output of the learnable frontend is fed to an attention-based model [Bahdanau et al., 2014], that contains one LSTM layer of hidden size 60 followed by an attention mechanism, inspired by Hsiao and Chen [2018]. The attention mechanism consists of two fully connected layers, of 50 and 1 unit respectively, and a softmax layer, that are applied to each output of the LSTM. The vector obtained is used to weight a linear combination of the LSTM outputs, that goes through another fully connected layer of size the number of labels considered. The detailed architecture is shown in Figure 8-2. In [Hsiao and Chen, 2018], this model reaches state-of-the-art performance when trained for emotion recognition on mel-filterbanks, which motivated using it for the paralinguistic task of dysarthria detection.

8.3 Experimental setup

We carry experiments on the TORGO database [Rudzicz et al., 2008]. It consists of sound recordings, sampled at 16kHz, from speakers with either cerebral palsy or amyotrophic lateral

Training	Validation	Test
FC02, F03 (VL), F01 (L), MC04, MC03, M02 (M)	MC02, FC01, M03 (VL), M01 (M)	FC03, F04 (VL), MC01, M05 (L), M04 (M)
3182 C, 1382 D	950 C, 802 D	2103 C, 997 D

Table 8.1: Speakers and number of recordings per set, the severity of each person is indicated after their ID: VL is Very Low, L is Low, and M is Medium. The bottom line shows the total number of Control (C) and Dysarthric (D) utterances per set.

sclerosis, which are two of the prevalent causes of dysarthria. Similar data for a control set of subjects is also available. Along with sound recordings, TORGO contains 3D articulatory features that we did not use.

There are five groups of people: the control group not affected by the disease, and 4 other groups of affected people, classified by the severity of the disease. Each person recorded has a code name, **F** is for female, **M** is for male, while **C** is for control, followed by an identification number. A random split of the database would result in similar speakers in training, validation, and test sets, that could reduce the task to a speaker identification task. To avoid this confounding factor, we split the database to have a good repartition of the different severities among the training, validation and test set, while having no common speakers between the different sets (see Table 8.1 for the detailed split).

After studying the database we decided to pad the recordings so they all last $2.5s$. We extract low level descriptors (LLDs) from it to have a first baseline. We use the OpenSmile toolkit [Eyben et al., 2010], with the configuration of the Interspeech 2009 Emotion Challenge [Schuller et al., 2009]. For each $25ms$ window of the recordings (strided by $10ms$), 32 features are extracted (12 MFCCs, root mean square energy, zero-crossing rate, harmonics-to-noise ratio, F_0 and their Δ).

Our second baseline takes as input mel-filterbanks. We pre-emphase the sound signals with a factor of 0.97. 64 mel-filterbanks are computed every 25ms with a stride of 10ms and passed through a log-compression. To evaluate Time-Domain filterbanks in a comparable setting, we design them with the same number of filters, window size and stride.

For the PCEN layer, we take $\epsilon = 10^{-6}$ and $s = 0.5$, both fixed. We initialize r, α and δ at 0.5, 0.98 and 2.0 respectively. During training, we constrain r to be non-negative. All

Input data	UAR % val.	UAR % test
LLDs	64.8 ± 1.2	65.5 ± 3.6
mel-filterbanks	79.9 ± 6.3	72.4 ± 3.0
mel-filterbanks + mvn	63.5 ± 1.7	70.3 ± 2.9
mel-filterbanks + PCEN	76.0 ± 6.1	79.7 ± 3.8
Time-Domain filterbanks	93.7 ± 1.2	82.4 ± 0.4

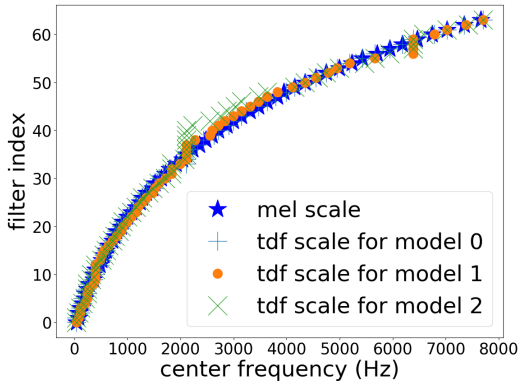
Table 8.2: UAR (%) of the attention-based model trained over different features or learnable frontends. The UAR is averaged over 3 runs and we report standard deviations.

models are trained with SGD with momentum (0.98) and batch size 1, with a learning rate of 0.001.

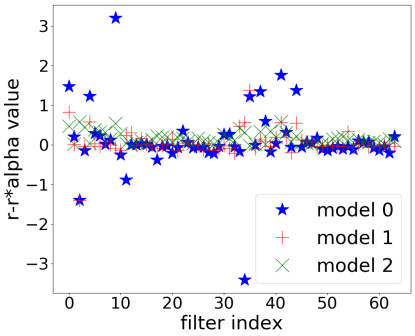
We use the Unweighted Average Recall (UAR) to evaluate our results. The UAR of a model is the mean of its accuracy for each label. It is a more informative metric when dealing with highly unbalanced datasets than the accuracy, since it is reweighting the results depending on the size of each class. It has been widely used in unbalanced settings such as the Emotion Recognition challenge [Schuller et al., 2009]. We use the validation set for hyperparameter selection and early stopping.

8.4 Results

Table 8.2 shows the UAR on the validation and test sets. All the results are the mean UAR obtained over three runs with different random initialization. We do not compare them to previously published results [Bhat et al., 2017, Kim et al., 2015] as they use additional data and/or perform a different task. The attention based-model trained on LLDs features reaches an accuracy of 66% and is our baseline system. Replacing LLDs by mel-filterbanks improves the performance by 6% in absolute. Adding a fixed mean-variance normalization step (mvn) brings the models to over-fitting, and thus the UAR decreases of 2%. However, we observe that replacing the fixed log-compression and mean-variance normalization step by a learnable PCEN layer improves the UAR of the models of 7% compared to the unnormalized mel-filterbanks. Moreover, an even bigger increase is noticed when replacing mel-filterbanks by equivalent Time-Domain filterbanks (10% in absolute). In this case,



(a) Center frequency of learnt filters



(b) Learnt compression exponents

Figure 8-3: Detailed analysis of filters and compression function learned by the model. Left shows the new scales obtained by three independent models using TD-filterbanks, compared to mel scale. The center frequency is the frequency for which a filter is maximum. Right shows an approximation of the compression exponents obtained for the PCEN layer learned on mel-filterbanks.

Time-Domain filterbanks are trained with a log-compression and no normalization. We can emphasize the fact that using the Time-Domain filterbanks also leads to a more stable learning process, as the standard deviation along different runs is considerably lower.

When studying the new scale learned by the TD-filterbanks (see Figure 8-3) (left)) we notice that the filters tend to focus around $2000Hz$ and $6500Hz$, which suggests that either those frequencies are crucial to identify dysarthria, or the model might exploit a bias in the dataset. Regardless, this illustrates why a non-linguistic classification task can benefit from learning its filter bank. Indeed, mel-filterbanks are very imprecise around $6500Hz$, and if a fine discriminative pattern were to be present in that region of the spectrum, mel-filterbanks would lose that information. However, Time-Domain filterbanks can learn to locate several, precise filters in any part of the spectrum. Similarly, in Figure 8-3 (right) we observe that the compression factor learned by PCEN varies between channels, unlike a log-compression which is applied equivalently to all channels.

8.4.1 Fully learnable frontend

As we observe independent gains from either learning the features or learning the compression-normalization, we explore in our final experiments learning jointly all these operations. We

Input data	UAR % val.	UAR % test
Time-Domain filterbanks + PCEN	72.3 ± 1.5	74.8 ± 1.1
Time-Domain filterbanks + PCEN only r	74.6 ± 2.9	76.4 ± 1.8
Time-Domain filterbanks + PCEN only α	66.6 ± 1.4	63.3 ± 8.2

Table 8.3: UAR (%) of the attention-based model trained over different fully learnable frontends. "Only r" means that only the compression factor of PCEN is learned, while "only" α" refers to the setting in which we only learn the normalization strength. When not specified, both components are learned. The UAR is averaged over 3 runs and standard deviations are reported.

remove the log-compression step of Time-Domain filterbanks and replace it by a PCEN layer. We use three settings: one for which r, α and δ are learned, the second one with only r (compression factor) learned, and finally the last one for which only α (normalization strength) is learned. If a parameter is not learned, it is fixed to its initial value (specified in Section 8.3). Table 8.3 shows that learning only the normalization exponent gives worse results than the models trained on LLDs. However, we notice that the model learning r, α and δ, and the one only learning r match the models using mel-filterbanks.

8.5 Conclusion

In this Chapter, and after positive results in speech recognition, we have shown that a paralinguistic task, dysarthria detection, also benefits significantly of learning the frontend with Time-Domain filterbanks. Moreover, we have proposed a fully learnable audio frontend, combining Time-Domain filterbanks and Per Channel Energy Normalization. It is the first time that a model is developed with the ability to learn the extraction, compression and normalization of the features from the raw waveform, jointly with a classifier. Learning only the Time-Domain filterbanks or the PCEN parameters gives better results than learning them jointly, but learning both still gives similar to better performance than using fixed features, which constitutes a proof of concept for fully learnable audio frontends.

Chapter 9

Conclusion

9.1 Summary of the contributions

Deep neural networks have changed the landscape of machine learning, from computer vision to reinforcement learning, natural language processing and speech recognition. The paradigm of deep learning has consistently been to replace formerly composite systems by end-to-end architectures, trained from the rawest form of the data up to the final target by backpropagation. In the case of speech recognition, acoustic models and then language models have been replaced by deep architectures, and the two-step training scheme of hybrid HMM-DNNs is now being outperformed by end-to-end training. However, along all this recent history of speech recognition, hardcoded speech features, mel-filterbanks in particular, have still not been made obsolete by learnable alternatives and remain a systematic component of state-of-the-art systems, both in recognition and paralinguistic classification.

The original topic of this thesis was the development of weakly-supervised and unsupervised algorithms for speech recognition. After preliminary studies in which we showed that weakly-supervised and unsupervised phonetic modelling could significantly benefit from using a richer deep scattering spectrum rather than mel-filterbanks, adressing the *statu quo* of using mel-filterbanks as input features became the central question of this thesis. Rather than exploring the space of exising fixed features, we decided to remove them and train directly deep neural networks from the raw waveform. We showed that we could disentangle speaker and phonetic information from the raw waveform with a single neural network, into embeddings that would generalize to other datasets or languages. We also showed that

learning from the waveform significantly improved the quality of speaker embeddings.

However, these first results on training models from the raw waveform were dampened by their poor performance on phonetic modelling, and by the difficulty to conduct fair and controlled experiments to compare against mel-filterbanks. Moreover, our exploration of architectures was mildly inspired, and lacked intuition on what was appropriate and important when training on the waveform. These observations led us to design a specific frontend, that can be plugged any neural architecture processing audio, rather than designing homogeneous neural networks from the waveform to the last layer, as we did previously. We introduced Time-Domain filterbanks, a lightweight neural architecture that processes raw speech, can be initialized as an approximation of mel-filterbanks, and then be trained jointly with any architecture. As the question of learning the frontend is orthogonal to the challenge of weakly-supervised learning, we decided to validate our approach on the most mainstream task that uses speech features: supervised speech recognition. Then, via extensive systematic studies and successive improvements, we showed that Time-Domain filterbanks consistently outperform mel-filterbanks in equivalent conditions (same acoustic model, number of filters, window size, window stride), as well as a previously proposed approach [Hoshen et al., 2015, Sainath et al., 2015a]. This is a noticeable achievement, as all our models (architecture, hyperparameters) were cross-validated on mel-filterbanks, and trained "as is" on Time-Domain filterbanks. This is the first time such a consistent improvement is observed. Finally, we integrated Time-Domain filterbanks into the first fully convolutional speech recognition system, which is currently the state-of-the-art among end-to-end system on both Wall Street Journal and Librispeech datasets.

These consistent results in speech recognition then led us to extend our approach to a paralinguistic classification, dysarthria detection from speech. We showed that Time-Domain filterbanks significantly outperform low-level descriptors and mel-filterbanks. We also used this task as a test bed to experiment with the first fully learnable frontend, that can learn jointly the feature extraction, a compression function, and a normalization.

9.2 Towards fully learnable audio understanding systems

So far, we have applied Time-Domain filterbanks to two tasks: speech recognition, and dysarthria detection. However, and as explained in Section 1.4, these are only few of a virtually unlimited number of classification tasks taking speech as input (see the Interspeech Computational Paralinguistics Challenge which introduces new tasks every year[1]), and still relying on fixed, handcrafted features (mel-filterbanks, MFCCs, or low-level descriptors). Comparing the relative improvement of Time-Domain filterbanks over mel-filterbanks for speech recognition and dysarthria detection is hard due to many confounding factors (task, performance of the baseline system, hyperparameter exploration), however the latter seems to benefit the most from learning the frontend rather than using fixed features. This is encouraging, as it confirms the intuition substantiated in 1.4 that paralinguistic tasks may benefit even more from learning from the raw waveform than speech recognition, for which mel-filterbanks have been tuned.

A rationale behind using Time-Domain filterbanks for dysarthria detection was that, since mel-filterbanks have been designed to mimick the human perception, they are undoubtedly suboptimal for tasks for which the human ear is not particularly tuned. This likely holds for many paralinguistics tasks, but maybe even more for non-speech audio classification tasks. Indeed, there has been a consistent use of mel-filterbanks for tasks such as audio event classification [Kons and Toledo-Ronen, 2013, Lim et al., 2016] (see the leaderboard of the DCASE2018 challenge[2]), birds vocalization detection [Lostanlen et al., 2018, Salamon et al., 2017], or whale call classification [Pace et al., 2010, Xian et al., 2015]. Mel-filterbanks are based on the premise that since our ear is less sensitive to variations in high frequencies than in low frequencies, we should have more precise descriptors for lower frequencies. This is valid for speech recognition, as speech production is meant to be intelligible to the human ear, and as such is adapted to its perceptual scale. Conversely, during the first year of their development, an infant will lose the ability to perceive phonetic contrasts that do not exist in their native language [Aslin, 1980].

A good illustration of the adaptation of perception to production is the case of Australian

[1] `http://www.compare.openaudio.eu/`

[2] `http://dcase.community/challenge2018/task-acoustic-scene-classification-results-a`

Aboriginals. Middle-ear infection has a very high prevalence in this population, leading to a hearing loss in up to 70% of the population [Coates et al., 2002], which affects their perception of the lower (below 500Hz) and upper (above 4000Hz) frequencies. Butcher et al. [2003] explain that this widespread hearing loss is a likely explanation for a linguistic phenomenon observed in Australian Aboriginal languages: a phonemic inventory which is unusually concentrated in the middle frequencies. This illustrates how speech production adapts to the limits of speech perception.

On the other hand, many informations conveyed by speech are not controlled to be intelligible to the ear, and as such would be poorly modelled by mel-filterbanks that mimick the auditory perception. In particular, any task that would need precise discrimination in high frequencies would suffer from using mel-filterbanks. A striking illustration of this limitation, made in section 8.4, is that dysarthria detection seems to rely on patterns in high frequencies (around $6500Hz$), a region of the spectrum where mel-filterbanks are very imprecise. One could learn an adapted filter bank from linear-scale spectrograms [Sainath et al., 2013], however spectrograms compute a convolution (with sinusoids) and a non-linearity (a squared modulus) that might was well be integrated into a neural network. This is why we think that Time-Domain filterbanks could also show improvements on many tasks that still use hardcoded features. Moreover, the Per Channel Energy Normalization (PCEN) [Wang et al., 2017] presented in section 8.2, has previously brought considerable improvements (over log compression and mean normalization) to aforementioned tasks, in particular bird vocalization detection, acoustic event classification, and scene classification [Lostanlen et al., 2019], as well as whale call classification[3]. Combining Time-Domain filterbanks and Per Channel Energy Normalization (or an equivalent) is still an open question, and we believe that if can combine both approaches into a fully learnable frontend, paralinguistic and non-speech tasks, or audio tasks in general, could be greatly improved. Some settings also imply preprocessing of the raw audio, before or jointly with the feature extraction, such as speech enhancement, source separation, or multi-channel beamforming. There has been previous attempts at integrating each of these steps into a neural architecture [Ochiai et al., 2017, Sainath et al., 2015b, Seki et al., 2018], however we are yet to see all of these operations, as well as feature extraction, compression and normalization being trained jointly for the task at hand.

[3] `https://ai.googleblog.com/2018/10/acoustic-detection-of-humpback-whales.html`

Task	Speech recognition [Young et al., 2002]	Music classification [Van den Oord et al., 2013]	Birds vocalization detection [Lostanlen et al., 2018]
Number of filters	40	128	40
Window size (ms)	25	23	12
Window stride (ms)	10	11.5	1.5

Table 9.1: Preferred hyperparameters of mel-filterbanks for three audio tasks.

9.3 Rethinking Time-Domain filterbanks: how much structure do we need?

Let us assume that we could integrate the beamforming, filtering, compression and normalization into a learnable frontend. This would allow learning all its steps, however its structure would still need to be defined beforehand. In particular, the structure of Time-Domain filterbanks contains several critical choices: the number and order of linear and non-linear layers, the choice of non-linearities, and the number of filters, window size, and window stride of each convolutional layer. As explained in Section 5.1, we chose these hyperparameters to match the standard 40 melfilterbanks, computed on 25ms windows, and strided by 10ms. The first motivation was that taking inspiration from the prominent speech features to design a learnable alternative was a promising avenue to finally outperform them. Furthermore, some hyperparameters, in particular the window size and stride (as well as the number of filters in chapters 5 and 6) were not explored beyond the standard parameters of mel-filterbanks to allow for fair comparisons and ablation studies. This also justifies why until Chapter 8, we only used a log-compression and did not learn the (optional) parameters of instance normalization that can be trained to modify the mean-variance normalization. Observing that in equivalent conditions, Time-Domain filterbanks consistently outperform their mel-filterbanks counterpart, we could then relax this constraint and explore more freely the space of architectures. In Chapter 7, we obtained our best results by doubling the number of filters compared to standard mel-filterbanks. This calls our current approach into question. We can draw a spectrum on the amount of structure and prior knowledge that are put into the audio frontend. On one end of this spectrum lie fixed, handcrafted speech features, based on psychoacoustics, signal processing and experimental knowledge. On the other end lie models that use generic neural network layers as a frontend [Hinton et al., 2012a, Palaz et al., 2013b].

We decided to strike a middle-ground by removing the bias in the parameters (by learning the convolutional weights, and optionally the compression and normalization), while putting a lot of prior knowledge into the structure of our neural network (pre-emphasis, complex convolutions and their size, squared modulus, low-pass filter and its stride, compression, normalization). In particular, assuming that we want to compute N filters on a window of size w and stride s, we hypothesized that we could reduce the bias of our system by learning these filters rather than using a mel-scale. However, choosing N, w and s could already induce a negative bias in our model. Table 9.1 shows preferred values for these hyperparameters for three tasks: speech recognition, music classification, and bird vocalization detection. It appears clearly that these hyperparameters are also task-dependent, and the same rationale that led us to learn the parameters of the frontend could also justify learning its structure. This would be the extreme end of a data-driven design of a frontend, rather than putting inductive bias from prior knowledge. This challenge can be addressed from the point of view of neural architecture search [Xie and Yuille, 2017, Zoph and Le, 2016], which applications to audio only amount (to the best of our knowledge) to first results on speech classification [Véniat et al., 2018, Wang et al., 2018, Zhang et al., 2018]. Moreover, all these few approaches use fixed mel-filterbanks, MFCCs or low-level descriptors, which leaves the question of frontend architecture search completely unexplored.

A radically opposite approach has been to put a lot of structure into a learnable frontend, to parametrize it with very few parameters, at the expense of a reduced expressivity. In particular, recent work has explored learning *sinc* functions [Ravanelli and Bengio, 2018] or spline wavelets [Balestriero et al., 2018] in the first layer. By constraining the class of filters the model can learn, it has been shown to learn efficiently with a fast convergence. We could have constrained Time-Domain filterbanks to explicitly learn analytic wavelets in the first layer, but we decided to use the Gabor wavelets only to initialize the weights, and then learn these weights like any other convolutional layer in the network.

We do not have a definitive answer to the question of finding the proper amount of structure constraints and prior knowledge to put into the design of a learnable frontend. A very constrained parametrization simplifies the optimization and can help in low-data regimes. However, we hypothesize that with the constant improvement in deep learning optimization techniques, as well as always bigger datasets, the horizon of learnable frontends will lie in always more data-driven approaches.

Appendix A

Fader Networks:Manipulating Images by Sliding Attributes

This appendix shows Fader Networks: Manipulating Images by Sliding Attributes Lample et al. [2017], a paper accepted for poster presentation at NIPS 2017, and a joint work with Guillaume Lample, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer and Marc'Aurelio Ranzato.

Fader Networks: Manipulating Images by Sliding Attributes

Guillaume Lample[1,2], **Neil Zeghidour**[1,3], **Nicolas Usunier**[1],
Antoine Bordes[1], **Ludovic Denoyer**[2], **Marc'Aurelio Ranzato**[1]
{gl,neilz,usunier,abordes,ranzato}@fb.com
ludovic.denoyer@lip6.fr

Abstract

This paper introduces a new encoder-decoder architecture that is trained to reconstruct images by disentangling the salient information of the image and the values of attributes directly in the latent space. As a result, after training, our model can generate different realistic versions of an input image by varying the attribute values. By using continuous attribute values, we can choose how much a specific attribute is perceivable in the generated image. This property could allow for applications where users can modify an image using sliding knobs, like *faders* on a mixing console, to change the facial expression of a portrait, or to update the color of some objects. Compared to the state-of-the-art which mostly relies on training adversarial networks in pixel space by altering attribute values at train time, our approach results in much simpler training schemes and nicely scales to multiple attributes. We present evidence that our model can significantly change the perceived value of the attributes while preserving the naturalness of images.

1 Introduction

We are interested in the problem of manipulating natural images by controlling some attributes of interest. For example, given a photograph of the face of a person described by their gender, age, and expression, we want to generate a realistic version of this same person looking older or happier, or an image of a hypothetical twin of the opposite gender. This task and the related problem of unsupervised domain transfer recently received a lot of interest [17, 24, 9, 26, 21, 23], as a case study for conditional generative models but also for applications like automatic image edition. The key challenge is that the transformations are ill-defined and training is unsupervised: the training set contains images annotated with the attributes of interest, but there is no example of the transformation: In many cases such as the "gender swapping" example above, there are no pairs of images representing the same person as a male or as a female. In other cases, collecting examples requires a costly annotation process, like taking pictures of the same person with and without glasses.

Our approach relies on an encoder-decoder architecture where, given an input image x with its attributes y, the encoder maps x to a latent representation z, and the decoder is trained to reconstruct x given (z, y). At inference time, a test image is encoded in the latent space, and the user chooses the attribute values y that are fed to the decoder. Even with binary attribute values at train time, each attribute can be considered as a continuous variable during inference to control how much it is perceived in the final image. We call our architecture *Fader Networks*, in analogy to the sliders of an audio mixing console, since the user can choose how much of each attribute they want to incorporate.

[1]Facebook AI Research
[2]Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6
[3]LSCP, ENS, EHESS, CNRS, PSL Research University, INRIA
Code available at https://github.com/facebookresearch/FaderNetworks

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Figure 1: Interpolation between different attributes (Zoom in for better resolution). Each line shows reconstructions of the same face with different attribute values, where each attribute is controlled as a continuous variable. It is then possible to make an old person look older or younger, a man look more manly or to imagine his female version. Left images are the originals.

The fundamental feature of our approach is to constrain the latent space to be invariant to the attributes of interest. Concretely, it means that the distribution over images of the latent representations should be identical for all possible attribute values. This invariance is obtained by using a procedure similar to domain-adversarial training (see e.g., [20, 6, 14]). In this process, a classifier learns to predict the attributes y given the latent representation z during training while the encoder-decoder is trained based on two objectives at the same time. The first objective is the reconstruction error of the decoder, i.e., the latent representation z must contain enough information to allow for the reconstruction of the input. The second objective consists in fooling the attribute classifier, i.e., the latent representation must prevent it from predicting the correct attribute values. In this model, achieving invariance is a means to filter out, or hide, the properties of the image that are related to the attributes of interest. A single latent representation thus corresponds to different images that share a common structure but with different attribute values. The reconstruction objective then forces the decoder to use the attribute values to choose, from the latent representation, the intended image.

Our motivation is to learn a disentangled latent space in which we have explicit control on some attributes of interest, without supervision of the intended result of modifying attribute values. With a similar motivation, several approaches have been tested on the same tasks [17, 24], on related image-to-image translation problems [9, 26], or for more specific applications like the creation of parametrized avatars [23]. In addition to a reconstruction loss, the vast majority of these works rely on adversarial training in pixel space, which compares during training images generated with an intentional change of attributes from genuine images for the target attribute values. Our approach is different both because we use adversarial training for the latent space instead of the output, but also because adversarial training aims at learning invariance to attributes. The assumption underlying our work is that a high fidelity to the input image is less conflicting with the invariance criterion, than with a criterion that forces the hallucinated image to match images from the training set.

As a consequence of this principle, our approach results in much simpler training pipelines than those based on adversarial training in pixel space, and is readily amenable to controlling multiple attributes, by adding new output variables to the discriminator of the latent space. As shown in Figure 1 on test images from the CelebA dataset [13], our model can make subtle changes to portraits that end up sufficient to alter the perceived value of attributes while preserving the natural aspect of the image and the identity of the person. Our experiments show that our model outperforms previous methods based on adversarial training on the decoders' output like [17] in terms of both reconstruction loss and generation quality as measured by human subjects. We believe this disentanglement approach is a serious competitor to the widespread adversarial losses on the decoder output for such tasks.

In the remainder of the paper, we discuss in more details the related work in Section 2. We then present the training procedure in Section 3 before describing the network architecture and the implementation in Section 4. Experimental results are shown in Section 5.

2 Related work

There is substantial literature on attribute-based and/or conditional image generation that can be split in terms of required supervision, with three different levels. At one extreme are fully supervised approaches developed to model known transformations, where examples take the form of *(input, transformation, result of the transformation)*. In that case, the model needs to learn the desired transformation. This setting was previously explored to learn affine transformations [8], 3D rotations [25], lighting variations [11] and 2D video game animations [19]. The methods developed in these works however rely on the supervised setting, and thus cannot be applied in our setup.

At the other extreme of the supervision spectrum lie fully unsupervised methods that aim at learning deep neural networks that disentangle the factors of variations in the data, without specification of the attributes. Example methods are InfoGAN [4], or the predictability minimization framework proposed in [20]. The neural photo editor [3] disentangles factors of variations in natural images for image edition. This setting is considerably harder than the one we consider, and it may be difficult with these methods to automatically discover high-level concepts such as gender or age.

Our work lies in between the two previous settings. It is related to information as in [15]. Methods developed for unsupervised domain transfer [9, 26, 21, 23] can also be applied in our case: given two different domains of images such as "drawings" and "photograph", one wants to map an image from one domain to the other without supervision; in our case, a domain would correspond to an attribute value. The mappings are trained using adversarial training in pixel space as mentioned in the introduction, using separate encoders and/or decoders per domain, and thus do not scale well to multiple attributes. In this line of work but more specifically considering the problem of modifying attributes, the Invertible conditional GAN [17] first trains a GAN conditioned on the attribute values, and in a second step learns to map input images to the latent space of the GAN, hence the name of invertible GANs. It is used as a baseline in our experiments. Antipov et al. [1] use a pre-trained face recognition system instead of a conditional GAN to learn the latent space, and only focuses on the age attribute. The attribute-to-image approach [24] is a variational auto-encoder that disentangles foreground and background to generate images using attribute values only. Conditional generation is performed by inferring the latent state given the correct attributes and then changing the attributes.

Additionally, our work is related to work on learning invariant latent spaces using adversarial training in domain adaptation [6], fair classification [5] and robust inference [14]. The training criterion we use for enforcing invariance is similar to the one used in those works, the difference is that the end-goal of these works is only to filter out nuisance variables or sensitive information. In our case, we learn generative models, and invariance is used as a means to force the decoder to use attribute information in its reconstruction.

Finally, for the application of automatically modifying faces using attributes, the feature interpolation approach of [22] presents a means to generate alterations of images based on attributes using a pre-trained network on ImageNet. While their approach is interesting from an application perspective, their inference is costly and since it relies on pre-trained models, cannot naturally incorporate factors or attributes that have not been foreseen during the pre-training.

3 Fader Networks

Let $\mathcal{X}$ be an image domain and $\mathcal{Y}$ the set of possible attributes associated with images in $\mathcal{X}$, where in the case of people's faces typical attributes are *glasses/no glasses*, *man/woman*, *young/old*. For simplicity, we consider here the case where attributes are binary, but our approach could be extended to categorical attributes. In that setting, $\mathcal{Y} = \{0, 1\}^n$, where n is the number of attributes. We have a training set $\mathcal{D} = \{(x^1, y^1), ..., (x^m, y^m)\}$, of m pairs (image, attribute) $(x^i \in \mathcal{X}, y^i \in \mathcal{Y})$. The end goal is to learn from $\mathcal{D}$ a model that will generate, for any attribute vector y', a version of an input image x whose attribute values correspond to y'.

Encoder-decoder architecture Our model, described in Figure 2, is based on an encoder-decoder architecture with domain-adversarial training on the latent space. The encoder $E_{\theta_{\text{enc}}} : \mathcal{X} \to \mathbb{R}^N$ is a convolutional neural network with parameters θ_{enc} that maps an input image to its N-dimensional latent representation $E_{\theta_{\text{enc}}}(x)$. The decoder $D_{\theta_{\text{dec}}} : (\mathbb{R}^N, \mathcal{Y}) \to \mathcal{X}$ is a deconvolutional network with parameters θ_{dec} that produces a new version of the input image given its latent representation $E_{\theta_{\text{enc}}}(x)$

and any attribute vector y'. When the context is clear, we simply use D and E to denote $D_{\theta_{\text{dec}}}$ and $E_{\theta_{\text{enc}}}$. The precise architectures of the neural networks are described in Section 4. The auto-encoding loss associated to this architecture is a classical mean squared error (MSE) that measures the quality of the reconstruction of a training input x given its true attribute vector y:

$$\mathcal{L}_{\text{AE}}(\theta_{\text{enc}}, \theta_{\text{dec}}) = \frac{1}{m} \sum_{(x,y)\in\mathcal{D}} \left\| D_{\theta_{\text{dec}}}\left(E_{\theta_{\text{enc}}}(x), y\right) - x \right\|_2^2$$

The exact choice of the reconstruction loss is not fundamental in our approach, and adversarial losses such as PatchGAN [12] could be used in addition to the MSE at this stage to obtain better textures or sharper images, as in [9]. Using a mean absolute or mean squared error is still necessary to ensure that the reconstruction matches the original image.

Ideally, modifying y in $D(E(x), y)$ would generate images with different perceived attributes, but similar to x in every other aspect. However, without additional constraints, the decoder learns to ignore the attributes, and modifying y at test time has no effect.

Learning attribute-invariant latent representations To avoid this behavior, our approach is to learn latent representations that are invariant with respect to the attributes. By invariance, we mean that given two versions of a same object x and x' that are the same up to their attribute values, for instance two images of the same person with and without glasses, the two latent representations $E(x)$ and $E(x')$ should be the same. When such an invariance is satisfied, the decoder must use the attribute to reconstruct the original image. Since the training set does not contain different versions of the same image, this constraint cannot be trivially added in the loss.

We hence propose to incorporate this constraint by doing adversarial training on the latent space. This idea is inspired by the work on predictability minimization [20] and adversarial training for domain adaptation [6, 14] where the objective is also to learn an invariant latent representation using an adversarial formulation of the learning objective. To that end, an additional neural network called the *discriminator* is trained to identify the true attributes y of a training pair (x, y) given $E(x)$. The invariance is obtained by learning the encoder E such that the discriminator is unable to identify the right attributes. As in GANs [7], this corresponds to a two-player game where the discriminator aims at maximizing its ability to identify attributes, and E aims at preventing it to be a good discriminator. The exact structure of our discriminator is described in Section 4.

Discriminator objective The discriminator outputs probabilities of an attribute vector $P_{\theta_{\text{dis}}}(y|E(x))$, where θ_{dis} are the discriminator's parameters. Using the subscript k to refer to the k-th attribute, we have $\log P_{\theta_{\text{dis}}}(y|E(x)) = \sum_{k=1}^{n} \log P_{\theta_{\text{dis}},k}(y_k|E(x))$. Since the objective of the discriminator is to predict the attributes of the input image given its latent representation, its loss depends on the current state of the encoder and is written as:

$$\mathcal{L}_{\text{dis}}(\theta_{\text{dis}}|\theta_{\text{enc}}) = -\frac{1}{m} \sum_{(x,y)\in\mathcal{D}} \log P_{\theta_{\text{dis}}}\left(y\middle|E_{\theta_{\text{enc}}}(x)\right) \tag{1}$$

Adversarial objective The objective of the encoder is now to compute a latent representation that optimizes two objectives. First, the decoder should be able to reconstruct x given $E(x)$ and y, and at the same time the discriminator should not be able to predict y given $E(x)$. We consider that a mistake is made when the discriminator predicts $1 - y_k$ for attribute k. Given the discriminator's parameters, the complete loss of the encoder-decoder architecture is then:

$$\mathcal{L}(\theta_{\text{enc}}, \theta_{\text{dec}}|\theta_{\text{dis}}) = \frac{1}{m} \sum_{(x,y)\in\mathcal{D}} \left\| D_{\theta_{\text{dec}}}\left(E_{\theta_{\text{enc}}}(x), y\right) - x \right\|_2^2 - \lambda_E \log P_{\theta_{\text{dis}}}(1 - y|E_{\theta_{\text{enc}}}(x)), \tag{2}$$

where $\lambda_E > 0$ controls the trade-off between the quality of the reconstruction and the invariance of the latent representations. Large values of λ_E will restrain the amount of information about x contained in $E(x)$, and result in blurry images, while low values limit the decoder's dependency on the latent code y and will result in poor effects when altering attributes.

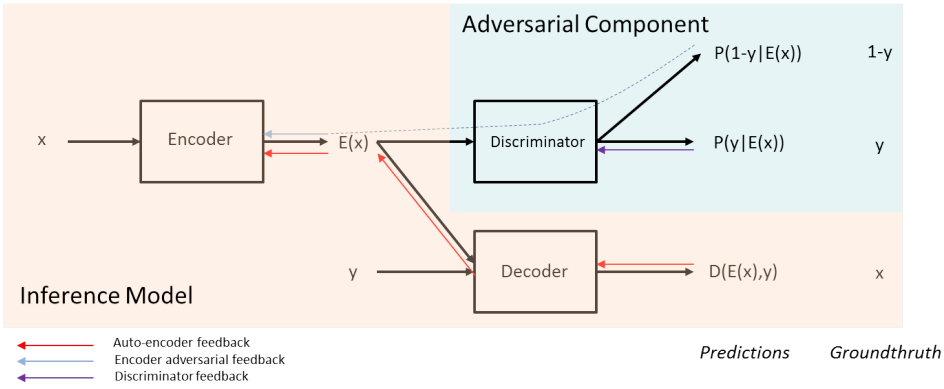


Figure 2: Main architecture. An (image, attribute) pair (x, y) is given as input. The encoder maps x to the latent representation z; the discriminator is trained to predict y given z whereas the encoder is trained to make it impossible for the discriminator to predict y given z only. The decoder should reconstruct x given (z, y). At test time, the discriminator is discarded and the model can generate different versions of x when fed with different attribute values.

Learning algorithm Overall, given the current state of the encoder, the optimal discriminator parameters satisfy $\theta^*_{\text{dis}}(\theta_{\text{enc}}) \in \text{argmin}_{\theta_{\text{dis}}} \mathcal{L}_{\text{dis}}(\theta_{\text{dis}}|\theta_{\text{enc}})$. If we ignore problems related to multiple (and local) minima, the overall objective function is

$$\theta^*_{\text{enc}}, \theta^*_{\text{dec}} = \underset{\theta_{\text{enc}}, \theta_{\text{dec}}}{\text{argmin}} \, \mathcal{L}(\theta_{\text{enc}}, \theta_{\text{dec}}|\theta^*_{\text{dis}}(\theta_{\text{enc}})) \, .$$

In practice, it is unreasonable to solve for $\theta^*_{\text{dis}}(\theta_{\text{enc}})$ at each update of θ_{enc}. Following the practice of adversarial training for deep networks, we use stochastic gradient updates for all parameters, considering the current value of θ_{dis} as an approximation for $\theta^*_{\text{dis}}(\theta_{\text{enc}})$. Given a training example (x, y), let us denote $\mathcal{L}_{\text{dis}}\big(\theta_{\text{dis}}\big|\theta_{\text{enc}}, x, y\big)$ the auto-encoder loss restricted to (x, y) and $\mathcal{L}\big(\theta_{\text{enc}}, \theta_{\text{dec}}\big|\theta_{\text{dis}}, x, y\big)$ the corresponding discriminator loss. The update at time t given the current parameters $\theta^{(t)}_{\text{dis}}$, $\theta^{(t)}_{\text{enc}}$, and $\theta^{(t)}_{\text{dec}}$ and the training example $(x^{(t)}, y^{(t)})$ is:

$$\begin{aligned}
\theta^{(t+1)}_{\text{dis}} &= \theta^{(t)}_{\text{dis}} - \eta \nabla_{\theta_{\text{dis}}} \mathcal{L}_{\text{dis}}\big(\theta^{(t)}_{\text{dis}}\big|\theta^{(t)}_{\text{enc}}, x^{(t)}, y^{(t)}\big) \\
[\theta^{(t+1)}_{\text{enc}}, \theta^{(t+1)}_{\text{dec}}] &= [\theta^{(t)}_{\text{enc}}, \theta^{(t)}_{\text{dec}}] - \eta \nabla_{\theta_{\text{enc}}, \theta_{\text{dec}}} \mathcal{L}\big(\theta^{(t)}_{\text{enc}}, \theta^{(t)}_{\text{dec}}\big|\theta^{(t+1)}_{\text{dis}}, x^{(t)}, y^{(t)}\big) \, .
\end{aligned}$$

The details of training and models are given in the next section.

4 Implementation

We adapt the architecture of our network from [9]. Let C_k be a Convolution-BatchNorm-ReLU layer with k filters. Convolutions use kernel of size 4×4, with a stride of 2, and a padding of 1, so that each layer of the encoder divides the size of its input by 2. We use leaky-ReLUs with a slope of 0.2 in the encoder, and simple ReLUs in the decoder.

The encoder consists of the following 7 layers:

$$C_{16} - C_{32} - C_{64} - C_{128} - C_{256} - C_{512} - C_{512}$$

Input images have a size of 256×256. As a result, the latent representation of an image consists of 512 feature maps of size 2×2. In our experiments, using 6 layers gave us similar results, while 8 layers significantly decreased the performance, even when using more feature maps in the latent state.

To provide the decoder with image attributes, we append the latent code to each layer given as input to the decoder, where the latent code of an image is the concatenation of the one-hot vectors representing

Model	Naturalness			Accuracy		
	Mouth	Smile	Glasses	Mouth	Smile	Glasses
Real Image	92.6	87.0	88.6	89.0	88.3	97.6
IcGAN AE	22.7	21.7	14.8	88.1	91.7	86.2
IcGAN Swap	11.4	22.9	9.6	10.1	9.9	47.5
FadNet AE	88.4	75.2	78.8	91.8	90.1	94.5
FadNet Swap	79.0	31.4	45.3	66.2	97.1	76.6

Table 1: Perceptual evaluation of naturalness and swap accuracy for each model. The naturalness score is the percentage of images that were labeled as "real" by human evaluators to the question "Is this image a real photograph or a fake generated by a graphics engine?". The accuracy score is the classification accuracy by human evaluators on the values of each attribute.

the values of its attributes (binary attributes are represented as $[1, 0]$ and $[0, 1]$). We append the latent code as additional constant input channels for all the convolutions of the decoder. Denoting by n the number of attributes, (hence a code of size $2n$), the decoder is symmetric to the encoder, but uses transposed convolutions for the up-sampling:

$$C_{512+2n} - C_{512+2n} - C_{256+2n} - C_{128+2n} - C_{64+2n} - C_{32+2n} - C_{16+2n} \, .$$

The discriminator is a C_{512} layer followed by a fully-connected neural network of two layers of size 512 and n repsectively.

Dropout We found it extremely beneficial to add dropout in our discriminator. We set the dropout rate to 0.3 in all our experiments. Following [9], we also tried to add dropout in the first layers of the decoder, but in our experiments, this turned out to significantly decrease the performance.

Discriminator cost scheduling Similarly to [2], we use a variable weight for the discriminator loss coefficient λ_E. We initially set λ_E to 0 and the model is trained like a normal auto-encoder. Then, λ_E is linearly increased to 0.0001 over the first $500,000$ iterations to slowly encourage the model to produce invariant representations. This scheduling turned out to be critical in our experiments. Without it, we observed that the encoder was too affected by the loss coming from the discriminator, even for low values of λ_E.

Model selection Model selection was first performed automatically using two criteria. First, we used the reconstruction error on original images as measured by the MSE. Second, we also want the model to properly swap the attributes of an image. For this second criterion, we train a classifier to predict image attributes. At the end of each epoch, we swap the attributes of each image in the validation set and measure how well the classifier performs on the decoded images. These two metrics were used to filter out potentially good models. The final model was selected based on human evaluation on images from the train set reconstructed with swapped attributes.

5 Experiments

5.1 Experiments on the celebA dataset

Experimental setup We first present experiments on the celebA dataset [13], which contains $200,000$ images of celebrity of shape 178×218 annotated with 40 attributes. We used the standard training, validation and test split. All pictures presented in the paper or used for evaluation have been taken from the test set. For pre-processing, we cropped images to 178×178, and resized them to 256×256, which is the resolution used in all figures of the paper. Image values were normalized to $[-1, 1]$. All models were trained with Adam [10], using a learning rate of 0.002, $\beta_1 = 0.5$, and a batch size of 32. We performed data augmentation by flipping horizontally images with a probability 0.5 at each iteration. As model baseline, we used IcGAN [17] with the model provided by the authors and trained on the same dataset. [4]

[4] `https://github.com/Guim3/IcGAN`

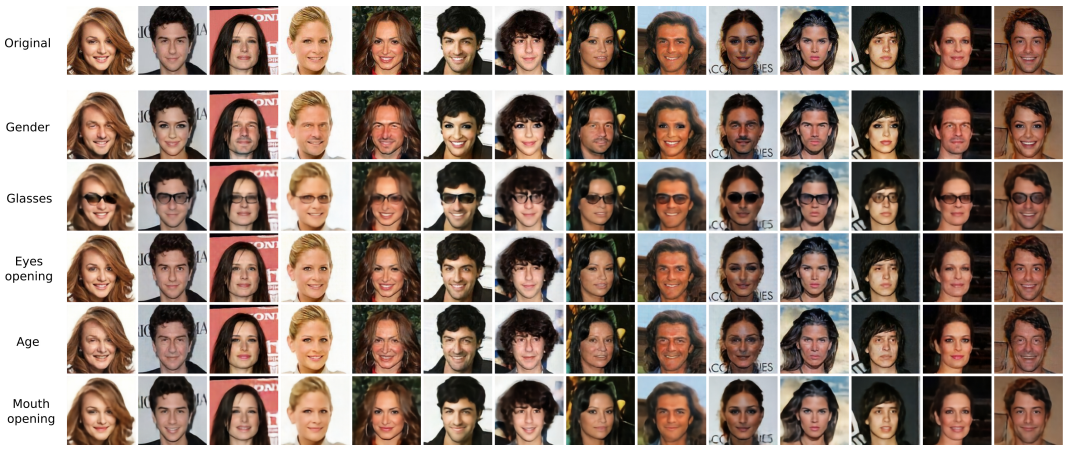


Figure 3: Swapping the attributes of different faces. Zoom in for better resolution.

Qualitative evaluation Figure 3 shows examples of images generated when swapping different attributes: the generated images have a high visual quality and clearly handle the attribute value changes, for example by adding realistic glasses to the different faces. These generated images confirm that the latent representation learned by Fader Networks is both invariant to the attribute values, but also captures the information needed to generate any version of a face, for any attribute value. Indeed, when looking at the shape of the generated glasses, different glasses shapes and colors have been integrated into the original face depending on the face: our model is not only adding "generic" glasses to all faces, but generates plausible glasses depending on the input.

Quantitative evaluation protocol We performed a quantitative evaluation of Fader Networks on Mechanical Turk, using IcGAN as a baseline. We chose the three attributes *Mouth (Open/Close)*, *Smile (With/Without)* and *Glasses (With/Without)* as they were attributes in common between IcGAN and our model. We evaluated two different aspects of the generated images: the **naturalness**, that measures the quality of generated images, and the **accuracy**, that measures how well swapping an attribute value is reflected in the generation. Both measures are necessary to assess that we generate natural images, and that the swap is effective. We compare: REAL IMAGE , that provides original images without transformation, FADNET AE and ICGAN AE , that reconstruct original images without attribute alteration, and FADNET SWAP and ICGAN SWAP , that generate images with one swapped attribute, e.g., *With Glasses* $\rightarrow$ *Without Glasses*. Before being submitted to Mechanical Turk, all images were cropped and resized following the same processing than IcGAN. As a result, output images were displayed in 64×64 resolution, also preventing Workers from basing their judgment on the sharpness of presented images exclusively.

Technically, we should also assess that the identity of a person is preserved when swapping attributes. This seemed to be a problem for GAN-based methods, but the reconstruction quality of our model is very good (RMSE on test of 0.0009, to be compared to 0.028 for IcGAN), and we did not observe this issue. Therefore, we did not evaluate this aspect.

For naturalness, the first 500 images from the test set such that there are 250 images for each attribute value were shown to Mechanical Turk Workers, 100 for each of the 5 different models presented above. For each image, we asked whether the image seems natural or generated. The description given to the Workers to understand their task showed 4 examples of real images, and 4 examples of fake images (1 FADNET AE , 1 FADNET SWAP , 1 ICGAN AE , 1 ICGAN SWAP).

The accuracy of each model on each attribute was evaluated in a different classification task, resulting in a total of 15 experiments. For example, the FadNet/Glasses experiment consisted in asking Workers whether people with glasses being added by FADNET SWAP effectively possess glasses, and vice-versa. This allows us to evaluate how perceptible the swaps are to the human eye. In each experiment, 100 images were shown (50 images per class, in the order they appear in the test set).

In both quantitative evaluations, each experiment was performed by 10 Workers, resulting in $5,000$ samples per experiment for naturalness, and $1,000$ samples per classification experiment on swapped attributes. The results on both tasks are shown in Table 1.

Figure 4: (Zoom in for better resolution.) Examples of multi-attribute swap (Gender / Opened eyes / Eye glasses) performed by the same model. Left images are the originals.

Quantitative results In the naturalness experiments, only around 90% of real images were classified as "real" by the Workers, indicating the high level of requirement to generate natural images. Our model obtained high naturalness accuracies when reconstructing images without swapping attributes: 88.4%, 75.2% and 78.8%, compared to IcGAN reconstructions whose accuracy does not exceed 23%, whether it be for reconstructed or swapped images. For the swap, FADNET SWAP still consistently outperforms ICGAN SWAP by a large margin. However, the naturalness accuracy varies a lot based on the swapped attribute: from 79.0% for the opening of the mouth, down to 31.4% for the smile.

Classification experiments show that reconstructions with FADNET AE and ICGAN AE have very high classification scores, and are even on par with real images on both Mouth and Smile. FADNET SWAP obtains an accuracy of 66.2% for the mouth, 76.6% for the glasses and 97.1% for the smile, indicating that our model can swap these attributes with a very high efficiency. On the other hand, with accuracies of 10.1%, 47.5% and 9.9% on these same attributes, ICGAN SWAP does not seem able to generate convincing swaps.

Multi-attributes swapping We present qualitative results for the ability of our model to swap multiple attributes at once in Figure 4, by jointly modifying the gender, open eyes and glasses attributes. Even in this more difficult setting, our model can generate convincing images with multiple swaps.

5.2 Experiments on Flowers dataset

We performed additional experiments on the Oxford-102 dataset, which contains about $9,000$ images of flowers classified into 102 categories [16]. Since the dataset does not contain other labels than the flower categories, we built a list of color attributes from the flower captions provided by [18]. Each flower is provided with 10 different captions. For a given color, we gave a flower the associated color attribute, if that color appears in at least 5 out of the 10 different captions. Although being naive, this approach was enough to create accurate labels. We resized images to 64×64. Figure 5 represents reconstructed flowers with different values of the "pink" attribute. We can observe that the color of the flower changes in the desired direction, while keeping the background cleanly unchanged.

Figure 5: Examples of reconstructed flowers with different values of the *pink* attribute. First row images are the originals. Increasing the value of that attribute will turn flower colors into pink, while decreasing it in images with originally pink flowers will make them turn yellow or orange.

6 Conclusion

We presented a new approach to generate variations of images by changing attribute values. The approach is based on enforcing the invariance of the latent space w.r.t. the attributes. A key advantage of our method compared to many recent models [26, 9] is that it generates realistic images of high resolution without needing to apply a GAN to the decoder output. As a result, it could easily be extended to other domains like speech, or text, where the backpropagation through the decoder can be really challenging because of the non-differentiable text generation process for instance. However, methods commonly used in vision to assess the visual quality of the generated images, like PatchGAN, could totally be applied on top of our model.

Acknowledgments

The authors would like to thank Yedid Hoshen for initial discussions about the core ideas of the paper, Christian Pursch and Alexander Miller for their help in setting up the experiments and Mechanical Turk evaluations. The authors are also grateful to David Lopez-Paz and Mouhamadou Moustapha Cisse for useful feedback and support on this project.

References

[1] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks. arXiv preprint arXiv:1702.01983, 2017.

[2] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. arXiv preprint arXiv:1511.06349, 2015.

[3] Andrew Brock, Theodore Lim, JM Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. arXiv preprint arXiv:1609.07093, 2016.

[4] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Advances in Neural Information Processing Systems, pages 2172–2180, 2016.

[5] Harrison Edwards and Amos Storkey. Censoring representations with an adversary. arXiv preprint arXiv:1511.05897, 2015.

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17(59):1–35, 2016.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.

[8] Geoffrey Hinton, Alex Krizhevsky, and Sida Wang. Transforming auto-encoders. Artificial Neural Networks and Machine Learning–ICANN 2011, pages 44–51, 2011.

[9] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. arXiv preprint arXiv:1611.07004, 2016.

[10] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[11] Tejas D Kulkarni, William F Whitney, Pushmeet Kohli, and Josh Tenenbaum. Deep convolutional inverse graphics network. In Advances in Neural Information Processing Systems, pages 2539–2547, 2015.

[12] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In European Conference on Computer Vision, pages 702–716. Springer, 2016.

[13] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV), 2015.

[14] Gilles Louppe, Michael Kagan, and Kyle Cranmer. Learning to pivot with adversarial networks. arXiv preprint arXiv:1611.01046, 2016.

[15] Michael F Mathieu, Junbo Jake Zhao, Junbo Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun. Disentangling factors of variation in deep representation using adversarial training. In Advances in Neural Information Processing Systems, pages 5041–5049, 2016.

[16] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on, pages 722–729. IEEE, 2008.

[17] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. arXiv preprint arXiv:1611.06355, 2016.

[18] Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. Learning deep representations of fine-grained visual descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 49–58, 2016.

[19] Scott E Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. In Advances in Neural Information Processing Systems, pages 1252–1260, 2015.

[20] Jürgen Schmidhuber. Learning factorial codes by predictability minimization. Neural Computation, 4(6):863–879, 1992.

[21] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. arXiv preprint arXiv:1611.02200, 2016.

[22] Paul Upchurch, Jacob Gardner, Kavita Bala, Robert Pless, Noah Snavely, and Kilian Weinberger. Deep feature interpolation for image content changes. arXiv preprint arXiv:1611.05507, 2016.

[23] Lior Wolf, Yaniv Taigman, and Adam Polyak. Unsupervised creation of parameterized avatars. arXiv preprint arXiv:1704.05693, 2017.

[24] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. Attribute2image: Conditional image generation from visual attributes. In European Conference on Computer Vision, pages 776–791. Springer, 2016.

[25] Jimei Yang, Scott E Reed, Ming-Hsuan Yang, and Honglak Lee. Weakly-supervised disentangling with recurrent transformations for 3d view synthesis. In Advances in Neural Information Processing Systems, pages 1099–1107, 2015.

[26] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv:1703.10593, 2017.

Appendix B

SING: Symbol-to-Instrument Neural Generator

This appendix shows SING: Symbol-to-Instrument Neural Generator Défossez et al. [2018], a paper accepted for poster presentation at NeurIPS 2018 (formerly NIPS), and a joint work with Alexandre Défossez, Nicolas Usunier, Léon Bottou and Francis Bach.

SING: Symbol-to-Instrument Neural Generator

Alexandre Défossez
Facebook AI Research
INRIA / ENS
PSL Research University
Paris, France
defossez@fb.com

Neil Zeghidour
Facebook AI Research
LSCP / ENS / EHESS / CNRS
INRIA / PSL Research University
Paris, France
neilz@fb.com

Nicolas Usunier
Facebook AI Research
Paris, France
usunier@fb.com

Léon Bottou
Facebook AI Research
New York, USA
leonb@fb.com

Francis Bach
INRIA
École Normale Supérieure
PSL Research University
francis.bach@ens.fr

Abstract

Recent progress in deep learning for audio synthesis opens the way to models that directly produce the waveform, shifting away from the traditional paradigm of relying on vocoders or MIDI synthesizers for speech or music generation. Despite their successes, current state-of-the-art neural audio synthesizers such as WaveNet and SampleRNN [24, 17] suffer from prohibitive training and inference times because they are based on autoregressive models that generate audio samples one at a time at a rate of 16kHz. In this work, we study the more computationally efficient alternative of generating the waveform frame-by-frame with large strides. We present SING, a lightweight neural audio synthesizer for the original task of generating musical notes given desired instrument, pitch and velocity. Our model is trained end-to-end to generate notes from nearly 1000 instruments with a single decoder, thanks to a new loss function that minimizes the distances between the log spectrograms of the generated and target waveforms. On the generalization task of synthesizing notes for pairs of pitch and instrument not seen during training, SING produces audio with significantly improved perceptual quality compared to a state-of-the-art autoencoder based on WaveNet [4] as measured by a Mean Opinion Score (MOS), and is about 32 times faster for training and $2,500$ times faster for inference.

1 Introduction

The recent progress in deep learning for sequence generation has led to the emergence of audio synthesis systems that directly generate the waveform, reaching state-of-the-art perceptual quality in speech synthesis, and promising results for music generation. This represents a shift of paradigm with respect to approaches that generate sequences of parameters to vocoders in text-to-speech systems [21, 23, 19], or MIDI partition in music generation [8, 3, 10]. A commonality between the state-of-the-art neural audio synthesis models is the use of discretized sample values, so that an audio sample is predicted by a categorical distribution trained with a classification loss [24, 17, 18, 14]. Another significant commonality is the use of autoregressive models that generate samples one-by-one, which leads to prohibitive training and inference times [24, 17], or requires specialized implementations and low-level code optimizations to run in real time [14]. An exception is parallel WaveNet [18] which generates a sequence with a fully convolutional network for faster inference. However, the parallel

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

approach is trained to reproduce the output of a standard WaveNet, which means that faster inference comes at the cost of increased training time.

In this paper, we study an alternative to both the modeling of audio samples as a categorical distribution and the autoregressive approach. We propose to generate the waveform for entire audio frames of 1024 samples at a time with a large stride, and model audio samples as continuous values. We develop and evaluate this method on the challenging task of generating musical notes based on the desired instrument, pitch, and velocity, using the large-scale NSynth dataset [4]. We obtain a lightweight synthesizer of musical notes composed of a 3-layer RNN with LSTM cells [12] that produces embeddings of audio frames given the desired instrument, pitch, velocity[1] and time index. These embeddings are decoded by a single four-layer convolutional network to generate notes from nearly 1000 instruments, 65 pitches per instrument on average and 5 velocities.

The successful end-to-end training of the synthesizer relies on two ingredients:

- A new loss function which we call the *spectral loss*, which computes the 1-norm between the log power spectrograms of the waveform generated by the model and the target waveform, where the power spectrograms are obtained by the short-time Fourier transform (STFT).

  Log power spectrograms are interesting both because they are related to human perception [6], but more importantly because the entire loss is invariant to the original phase of the signal, which can be arbitrary without audible differences.
- Initialization with a pre-trained autoencoder: a purely convolutional autoencoder architecture on raw waveforms is first trained with the spectral loss. The LSTM is then initialized to reproduce the embeddings given by the encoder, using mean squared error. After initialization, the LSTM and the decoder are fine-tuned together, backpropagating through the spectral loss.

We evaluate our synthesizer on a new task of pitch completion: generating notes for pitches not seen during training. We perform perceptual experiments with human evaluators to aggregate a Mean Opinion Score (MOS) that characterizes the naturalness and appealing of the generated sounds. We also perform ABX tests to measure the relative similarity of the synthesizer's ability to effectively produce a new pitch for a given instrument, see Section 5.3.2. We use a state-of-the-art autoencoder of musical notes based on WaveNet [4] as a baseline neural audio synthesis system. Our synthesizer achieves higher perceptual quality than Wavenet-based autoencoder in terms of MOS and similarity to the ground-truth while being about 32 times faster during training and $2,500$ times for generation.

2 Related Work

A large body of work in machine learning for audio synthesis focuses on generating parameters for vocoders in speech processing [21, 23, 19] or musical instrument synthesizers in automatic music composition [8, 3, 10]. Our goal is to learn the synthesizers for musical instruments, so we focus here on methods that generate sound without calling such synthesizers.

A first type of approaches model power spectrograms given by the STFT [4, 9, 25], and generate the waveform through a post-processing that is not part of the training using a phase reconstruction algorithm such as the Griffin-Lim algorithm [7]. The advantage is to focus on a distance between high-level representations that is more relevant perceptually than a regression on the waveform. However, using Griffin-Lim means that the training is not end to end. Indeed the predicted spectrograms may not come from a real signal. In that case, Griffin-Lim performs an orthogonal projection onto the set of valid spectrograms that is not accounted for during training. Notice that our approach with the spectral loss is different: our models directly predict waveforms rather than spectrograms and the spectral loss computes log power spectrograms of these predicted waveforms.

The current state-of-the-art in neural audio synthesis is to generate directly the waveform [24, 17, 19]. Individual audio samples are modeled with a categorical distribution trained with a multiclass cross-entropy loss. Quantization of the 16 bit audio is performed (either linear [17] or with a μ-law companding [24]) to map to a few hundred bins to improve scalability. The generation is still extremely costly; distillation [11] to a faster model has been proposed to reduce inference time at the expense of an even larger training time [18]. The recent proposal of [14] partly solves the issue with

[1] Quoting [4]: "MIDI velocity is similar to volume control and they have a direct relationship. For physical intuition, higher velocity corresponds to pressing a piano key harder."

a small loss in accuracy, but it requires heavy low-level code optimization. In contrast, our approach trains and generate waveforms comparably fast with a PyTorch[2] implementation. Our approach is different since we model the waveform as a continuous signal and use the spectral loss between generated and target waveforms and model audio frames of 1024 samples, rather than performing classification on individual samples. The spectral loss we introduce is also different from the power loss regularization of [18], even though both are based on the STFT of the generated and target waveforms. In [18], the primary loss is the classification of individual samples, and their power loss is used to equalize the average amplitude of frequencies over time. Thus the power loss cannot be used alone to learn to reconstruct the waveform.

Works on neural audio synthesis conditioned on symbolic inputs were developed mostly for text-to-speech synthesis [24, 17, 25]. Experiments on generation of musical tracks based on desired properties were described in [24], but no systematic evaluation has been published. The model of [4], which we use as baseline in our experiments on perceptual quality, is an autoencoder of musical notes based on WaveNet [24] that compresses the signal to generate high-level representations that transfer to music classification tasks, but contrarily to our synthesizer, it cannot be used to generate waveforms from desired properties of the instrument, pitch and velocity without some input signal.

The minimization by gradient descent of an objective function based on the power spectrogram has already been applied to the transformation of a white noise waveform into a specific sound texture [2]. However, to the best of our knowledge, such objective functions have not been used in the context of neural audio synthesis.

3 The spectral loss for waveform synthesis

Previous work in audio synthesis on the waveform focused on classification losses [17, 24, 4]. However, their computational cost needs to be mitigated by quantization, which inherently limits the resolution of the predictions, and ultimately increasing the number of classes is likely necessary to achieve the optimal accuracy. Our approach directly predicts a single continuous value for each audio sample, and computes distances between waveforms in the domain of power spectra to be invariant to the original phase of the signal. As a baseline, we also consider computing distances between waveforms using plain mean square error (MSE).

3.1 Mean square regression on the waveform

The simplest way of measuring the distance between a reconstructed signal $\hat{x}$ and the reference x is to compute the MSE on the waveform directly, that is taking the Euclidean norm between x and $\hat{x}$,

$$\mathrm{L}_{\mathrm{wav}}\left(x, \hat{x}\right) := \left\|x - \hat{x}\right\|^2. \tag{3.1}$$

The MSE is most likely not suited as a perceptual distance between waveforms because it is extremely sensitive to a small shift in the signal. Yet, we observed that it was sufficient to learn an autoencoder and use it as a baseline.

3.2 Spectral loss

As an alternative to the MSE on the waveform, we suggest taking the Short Term Fourier Transform (STFT) of both x and $\hat{x}$ and compare their absolute values in log scale. We first compute the log spectrogram

$$l(x) := \log\left(\epsilon + \left|\mathrm{STFT}\left[x\right]\right|^2\right). \tag{3.2}$$

The STFT decomposes the original signal x in successive frames of 1024 time steps with a stride of 256 so that a frame overlaps at 75% with the next one. The output for a single frame is given by 513 complex numbers, each representing a specific frequency range. Taking the point-wise absolute values of those numbers represents how much energy is present in a specific frequency range. We observed that our models generated higher quality sounds when trained using a log scale of those coefficients. Previous work has come to the same conclusion [4]. We observed that many entries

[2] `https://pytorch.org/`

of the spectrograms are close to zero and that small errors on those parts can add up to form noisy artifacts. In order to favor sparsity in the spectrogram, we use the $\|\cdot\|_1$ norm instead of the MSE,

$$\mathrm{L}_{\mathrm{stft},1}\left(x, \hat{x}\right) := \left\|l(x) - l(\hat{x})\right\|_1 . \tag{3.3}$$

The value of ϵ controls the trade-off between accurately representing low energy and high energy coefficients in the spectrogram. We found that $\epsilon = 1$ gave the best subjective reconstruction quality.

The STFT is a (complex) convolution operator on the waveform and the squared absolute value of the Fourier coefficients makes the power spectrum differentiable with respect to the generated waveform. Since the generated waveform is itself a differentiable function of the parameters (up to the non-differentiability points of activation functions such as ReLU), the spectral loss (3.3) can be minimized by standard backpropagation. Even though we only consider this spectral loss in our experiments, alternatives to the STFT such as the Wavelet transform also define differentiable loss for suitable wavelets.

3.2.1 Non unicity of the waveform representation

To illustrate the importance of the spectral loss instead of a waveform loss, let us now consider a problem that arises when generating notes in the test set. Let us assume one of the instrument is a pure sinuoide. For a given pitch at a frequency f, the audio signal is $x_i = \sin(2\pi i \frac{f}{16000} + \phi)$. Our perception of the signal is not affected by the choice of $\phi \in [0, 2\pi[$, and the power spectrogram of x is also unaltered. When recording an acoustic instrument, the value of ϕ depends on any number of variables characterizing the physical system that generated the sound and there is no guarantee that ϕ stays constant when playing the same note again. For a synthetic sound, ϕ also depends on implementation details of the software generating the sound.

For a sound that is not in the training set and as far as the model is concerned, ϕ is a random variable that can take any value in the range $[0, 2\pi[$. As a result, x_0 is unpredictable in the range $[-1, 1]$, and the mean square error between the generated signal and the ground truth is uninformative. Even on the training dataset, the model has to use extra resources to remember the value of ϕ for each pitch. We believe that this phenomenon is the reason why training the synthesizer using the MSE on the waveform leads to worse reconstruction performance, even though this loss is sufficient in the context of auto-encoding (see Section 5.2). The spectral loss solves this issue since the model is free to choose a single canonical value for ϕ.

However, one should note that the spectral loss is permissive, in the sense that it does not penalize phase inconsitencies of the complex phase across the different frames of the STFT, which lead to potential artifacts. In practice, we obtain state of the art results (see Section 5) and we conjecture that thanks to the frame overlap in the STFT, the solution that minimizes the spectral loss will often be phase consistent, which is why Griffin-Lim works resonably well despite sharing the same limitation.

4 Model

In this section we introduce the SING architecture. It is composed of two parts: a LSTM based sequence generator whose output is plugged to a decoder that transforms it into a waveform. The model is trained to recover a waveform x sampled at 16,000 Hz from the training set based on the one-hot encoded instrument I, pitch P and velocity V. The whole architecture is summarized in Figure 1.

4.1 LSTM sequence generator

The sequence generator is composed of a 3-layer recurrent neural network with LSTM cells and 1024 hidden units each. Given an example with velocity V, instrument I and pitch P, we obtain 3 embeddings $(u_V, v_I, w_P) \in \mathbb{R}^2 \times \mathbb{R}^{16} \times \mathbb{R}^8$ from look-up tables that are trained along with the model. Furthermore, the model is provided at each time step with an extra embedding $z_T \in \mathbb{R}^4$ where T is the current time step [22, 5], also obtained from a look-up table that is trained jointly. The input of the LSTM is the concatenation of those four vectors (u_V, v_I, w_P, z_T). Although we first experimented with an autoregressive model where the previous output was concatenated with those embeddings, we achieved much better performance and faster training by feeding the LSTM with only on the 4 vectors (u_V, v_I, w_P, z_T) at each time step. Given those inputs, the recurrent network

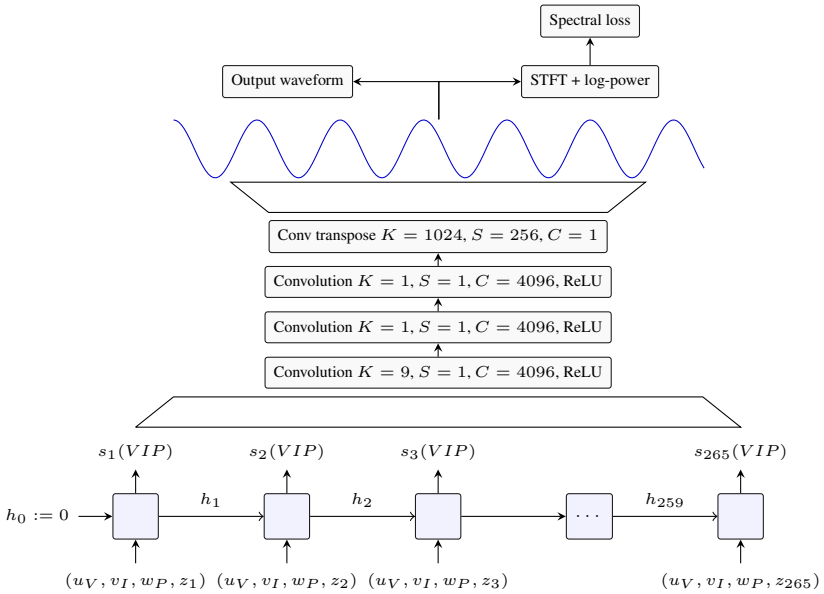


Figure 1: Summary of the entire architecture of SING. u_V, v_I, w_P, z_* represent the look-up tables respectively for the velocity, instrument, pitch and time. h_* represent the hidden state of the LSTM and s_* its output. For convolutional layers, K represents the kernel size, S the stride and C the number of channels.

generates a sequence $\forall 1 \leq T \leq N, s(V, I, P)_T \in \mathbb{R}^D$ with a linear layer on top of the last hidden state. In our experiments, we have $D = 128$ and $N = 265$.

4.2 Convolutional decoder

The sequence $s(V, I, P)$ is decoded into a waveform by a convolutional network. The first layer is a convolution with a kernel size of 9 and a stride of 1 over the sequence s with 4096 channels followed by a ReLU. The second and third layers are both convolutions with a kernel size of 1 (a.k.a. 1x1 convolution [4]) also followed by a ReLU. The number of channels is kept at 4096. Finally the last layer is a transposed convolution with a stride of 256 and a kernel size of 1024 that directly outputs the final waveform corresponding to an audio frame of size 1024. In order to reduce artifacts generated by the high stride value, we smooth the deconvolution filters by multiplying them with a squared Hann window. As the stride is one fourth of the kernel size, the squared Hann window has the property that the sum of its values for a given output position is always equal to 1 [7]. Thus the final deconvolution can also be seen as an overlap-add method. We pad the examples so that the final generated audio signal has the right length. Given our parameters, we need $s(V, I, P)$ to be of length $N = 265$ to recover a 4 seconds signal $d(s(V, I, P)) \in \mathbb{R}^{64,000}$.

4.3 Training details

All the models are trained on 4 P100 GPUs using Adam [15] with a learning rate of 0.0003 and a batch size of 256.

Initialization with an autoencoder. We introduce an encoder turning a waveform x into a sequence $e(x) \in \mathbb{R}^{N \times D}$. This encoder is almost the mirror of the decoder. It starts with a convolution layer with a kernel size of 1024, a stride of 256 and 4096 channels followed by a ReLU. Similarly to the decoder, we smooth its filters using a squared Hann window. Next are two 1x1 convolutions with 4096 channels and ReLU as an activation function. A final 1x1 convolution with no non linearity turns those 4096 channels into the desired sequence with D channels. We first train the encoder and decoder together as an auto-encoder on a reconstruction task. We train the auto-encoder for 50 epochs which takes about 12 hours on 4 GPUs.

LSTM training. Once the auto-encoder has converged, we use the encoder to generate a target sequence for the LSTM. We use the MSE between the output $s(V, I, P)$ of the LSTM and the output $e(x)$ of the encoder, only optimizing the LSTM while keeping the encoder constant. The LSTM is trained for 50 epochs using truncated backpropagation through time [26] using a sequence length of 32. This takes about 10 hours on 4 GPUs.

End-to-end fine tuning. We then plug the decoder on top of the LSTM and fine tune them together in an end-to-end fashion, directly optimizing for the loss on the waveform, either using the MSE on the waveform or computing the MSE on the log-amplitude spectrograms and back propagating through the STFT. At that point we stop using truncated back propagation through time and directly compute the gradient on the entire sequence. We do so for 20 epochs which takes about 8 hours on 4 GPUs. From start to finish, SING takes about 30 hours on 4 GPUs to train.

Although we could have initialized our LSTM and decoder randomly and trained end-to-end, we did not achieve convergence until we implemented our initialization strategy.

5 Experiments

The source code for SING and a pretrained model are available on our github[3]. Audio samples are available on the article webpage[4].

5.1 NSynth dataset

The train set from the NSynth dataset [4] is composed of 289,205 audio recordings of instruments, some synthetic and some acoustic. Each recording is 4 second long at 16,000 Hz and is represented by a vector $x_{V,I,P} \in [-1, 1]^{64,000}$ indexed by $V \in \{0, 4\}$ representing the velocity of the note, $I \in \{0, \dots, 1005\}$ representing the instrument, $P \in \{0, \dots, 120\}$ representing the pitch. The range of pitches available can vary depending on the instrument but for any combination of V, I, P, there is at most a single recording.

We did not make use of the validation or test set from the original NSynth dataset because the instruments had no overlap with the training set. Because we use a look-up table for the instrument embedding, we cannot generate audio for unseen instruments. Instead, we selected for each instrument 10% of the pitches randomly that we moved to a separate test set. Because the pitches are different for each instrument, our model trains on all pitches but not on all combinations of a pitch and an instrument. We can then evaluate the ability of our model to generalize to unseen combinations of instrument and pitch. In the rest of the paper, we refer to this new split of the original train set as the train and test set.

5.2 Generalization through pitch completion

We report our results in Table 1. We provided both the performance of the complete model as well as that of the autoencoder used for the initial training of SING. This autoencoder serves as a reference for the maximum quality the model can achieve if the LSTM were to reconstruct perfectly the sequence $e(x)$.

Although using the MSE on the waveform works well as far as the autoencoder is concerned, this loss is hard to optimize for the LSTM. Indeed, the autoencoder has access to the signal it must reconstruct, so that it can easily choose which *representation* of the signal to output as explained in Section 3.2.1. SING must be able to recover that information solely from the embeddings given to it as input. It manages to learn some of it but there is an important drop in quality. Besides, when switching to the test set one can see that the MSE on the waveform increases significantly. As the model has never seen those examples, it has no way of picking the right representation. When using a spectral loss, SING is free to choose a *canonical* representation for the signal it has to reconstruct and it does not have to remember the one that was in the training set. We observe that although we have a drop in quality between the train and test set, our model is still able to generalize to unseen combinations of pitch and instrument.

[3] `https://github.com/facebookresearch/SING`

[4] `https://research.fb.com/publications/sing-symbol-to-instrument-neural-generator`

		Spectral loss		Wav MSE	
Model	**training loss**	*train*	*test*	*train*	*test*
Autoencoder	waveform	0.026	0.028	0.0002	0.0003
SING	waveform	0.075	0.084	0.006	0.039
Autoencoder	spectral	0.028	0.032	N/A	N/A
SING	spectral	0.039	0.051	N/A	N/A
SING no time embedding	spectral	0.050	0.063	N/A	N/A

Table 1: Results on the train and test set of the pitch completion task for different models. The first column specifies the model, either the autoencoder used for the initial training of the LSTM or the complete SING model with the LSTM and the convolutional decoder. We compare models either trained with a loss on the waveform (see (3.1)) or on the spectrograms (see (3.3)). Finally we also trained a model with no temporal embedding.

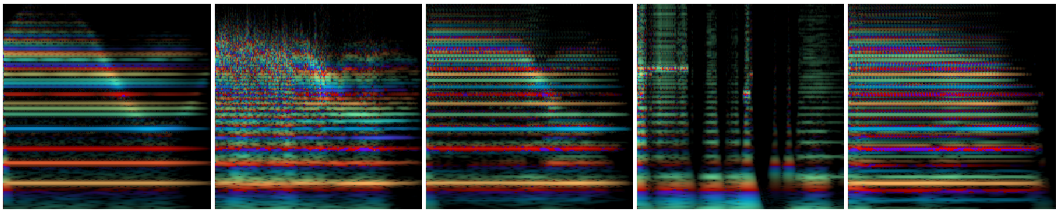


Figure 2: Example of rainbowgrams from the NSynth dataset and the reconstructions by different models. Rainbowgrams are defined in [4] as "a CQT spectrogram with intensity of lines proportional to the log magnitude of the power spectrum and color given by the derivative of the phase". Time is represented on the horizontal axis while frequencies are on the vertical one. From left to right: ground truth, Wavenet-based autoencoder, SING with spectral loss, SING with waveform loss and SING without the time embedding.

Finally, we tried training a model without the time embedding z_T. Theoretically, the LSTM could do without it by learning to count the number of time steps since the beginning of the sequence. However we do observe a significant drop in performance when removing this embedding, thus motivating our choice.

On Figure 2, we represented the rainbowgrams for a particular example from the test set as well as its reconstruction by the Wavenet-autoencoder, SING trained with the spectral and waveform loss and SING without time embedding. Rainbowgrams are defined in [4] as "a CQT spectrogram with intensity of lines proportional to the log magnitude of the power spectrum and color given by the derivative of the phase". A different derivative of the phase will lead to audible deformations of the target signal. Such modification are not penalized by our spectral loss as explained in Section 3.2.1. Nevertheless, we observe a mostly correct reconstruction of the derivative of the phase using SING. More examples from the test set, including the rainbowgrams and audio files are available on the article webpage[5].

5.3 Human evaluations

During training, we use several automatic criteria to evaluate and select our models. These criteria include the MSE on spectrograms, magnitude spectra, or waveform, and other perceptually-motivated metrics such as the Itakura-Saito divergence [13]. However, the correlation of these metrics with human perception remains imperfect, this is why we use human judgments as a metric of comparison between SING and the Wavenet baseline from [4].

[5] `https://research.fb.com/publications/sing-symbol-to-instrument-neural-generator`

Model	MOS	Training time (hrs * GPU)	Generation speed	Compression factor	Model size
Ground Truth	3.86 ± 0.24	-	-	-	-
Wavenet	2.85 ± 0.24	3840^*	0.2 sec/sec	32	948 MB
SING	3.55 ± 0.23	120	512 sec/sec	2133	243 MB

Table 2: Mean Opinion Score (MOS) and computational load of the different models. The training time is expressed in hours * GPU units, the generation time is expressed as the number of seconds of audio that can be generated per second of processing time. The compression factor represents the ratio between the dimensionality of the audio sequences ($64,000$ values) and either the latent state of Wavenet or the input vectors to SING. We also report the size of the models, in MB.
$(^*)$ Time corrected to account for the difference in FLOPs of the GPUs used.

5.3.1 Evaluation of perceptual quality: Mean Opinion Score

The first characteristic that we want to measure from our generated samples is their naturalness: how good they sound to the human ear. To do so, we perform experiments on Amazon Mechanical Turk [1] to get a Mean Opinion Score for the ground truth samples, and for the waveforms generated by SING and the Wavenet baseline. We did not include a Griffin-Lim based baseline as the authors in [4] concluded to the superiority of their Wavenet autoencoder.

We randomly select 100 examples from our test set. For the Wavenet-autoencoder, we pass these 100 examples through the network and retrieve the output. The latter is a pre-trained model provided by the authors of [4][6]. Notice that all of the 100 samples were used for training of the Wavenet-autoencoder, while they were not seen during the training of our models. For SING, we feed it the instrument, pitch and velocity information of each of the 100 samples. Workers are asked to rate the quality of the samples on a scale from 1 ("Very annoying and objectionable distortion. Totally silent audio") to 5 ("Imperceptible distortion"). Each of the 300 samples (100 samples per model) is evaluated by 60 Workers. The quality of the hardware used by Workers being variable, this could impede the interpretability of the results. Thus, we use the crowdMOS toolkit [20] which detects and discards inaccurate scores. This toolkit also allows to only keep the evaluations that are made with headphones (rather than laptop speakers for example), and we choose to do so as good listening conditions are necessary to ensure the validity of our measures. We report the Mean Opinion Score for the ground-truth audio and each of the 2 models in Table 2, along with the 95% confidence interval.

We observe that SING shows a significantly better MOS than the Wavenet-autoencoder baseline despite a compression factor which is 66 times higher. Moreover, to spotlight the benefits of our approach compared to the Wavenet baseline, we also report three metrics to quantify the computational load of the different models. The first metric is the training time, expressed in hours multiplied by the number of GPUs. The authors of [4], mention that their model trains for 10 days on 32 GPUs, which amounts to 7680 hours*GPUs. However, the GPUs used are capable of about half the FLOPs compared to our P100. Therefore, we corrected this value to 3840 hours*GPUs. On the other hand, SING is trained in 30 hours on four P100, which is 32 times faster than Wavenet. A major drawback of autoregressive models such as Wavenet is that the generation process is inherently sequential: generating the sample at time $t+1$ takes as input the sample at time t. We timed the generation using the implementation of the Wavenet-autoencoder provided by the authors, in its *fastgen* version[7] which is significantly faster than the original model. This yields a 22 minutes time to generate a 4-second sample. On a single P100 GPU, Wavenet can generate up to 64 sequences at the same time before reaching the memory limit, which amounts to 0.2 seconds of audio generated per second. On the other hand, SING can generate 512 seconds of audio per second of processing time, and is thus 2500 times faster than Wavenet. Finally, SING is also efficient in memory compared to Wavenet, as the model size in MB is more than 4 times smaller than the baseline.

5.3.2 ABX similarity measure

Besides absolute audio quality of the samples, we also want to ensure that when we condition SING on a chosen combination of instrument, pitch and velocity, we generate a relevant audio sample. To

[6] https://github.com/tensorflow/magenta/tree/master/magenta/models/nsynth

[7] https://magenta.tensorflow.org/nsynth-fastgen

do so, we measure how close samples generated by SING are to the ground-truth relatively to the Wavenet baseline. This measure is made by performing ABX [16] experiments: the Worker is given a ground-truth sample as a reference. Then, they are presented with the corresponding samples of SING and Wavenet, in a random order to avoid bias and with the possibility of listening as many times to the samples as necessary. They are asked to pick the sample which is the closest to the reference according to their judgment. We perform this experiment on 100 ABX triplets made from the same data as for the MOS, each triplet being evaluated by 10 Workers. On average over 1000 ABX tests, 69.7% are in favor of SING over Wavenet, which shows a higher similarity between our generated samples and the target musical notes than Wavenet.

Conclusion

We introduced a simple model architecture, SING, based on LSTM and convolutional layers to generate waveforms. We achieve state-of-the-art results as measured by human evaluation on the NSynth dataset for a fraction of the training and generation cost of existing methods. We introduced a spectral loss on the generated waveform as a way of using time-frequency based metrics without requiring a post-processing step to recover the phase of a power spectrogram. We experimentally validated that SING was able to embed music notes into a small dimension vector space where the pitch, instrument and velocity were disentangled when trained with this spectral loss, as well as synthesizing pairs of instruments and pitches that were not present in the training set. We believe SING opens up new opportunities for lightweight quality audio synthesis with potential applications for speech synthesis and music generation.

Acknowledgments

Authors thank Adam Polyak for his help on setting up the human evaluations and Axel Roebel for the insightful discussions. We also thank the Magenta team for their inspiring work on NSynth.

References

[1] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on psychological science*, 6(1):3–5, 2011.

[2] Hugo Caracalla and Axel Roebel. Gradient conversion between time and frequency domains using wirtinger calculus. In *DAFx 2017*, 2017.

[3] Kemal Ebcioğlu. An expert system for harmonizing four-part chorales. *Computer Music Journal*, 12(3):43–51, 1988.

[4] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. Neural audio synthesis of musical notes with wavenet autoencoders. Technical Report 1704.01279, arXiv, 2017.

[5] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*, 2017.

[6] JL Goldstein. Auditory nonlinearity. *The Journal of the Acoustical Society of America*, 41(3):676–699, 1967.

[7] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984.

[8] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. Deepbach: a steerable model for bach chorales generation. Technical Report 1612.01010, arXiv, 2016.

[9] Albert Haque, Michelle Guo, and Prateek Verma. Conditional end-to-end audio transforms. Technical Report 1804.00047, arXiv, 2018.

[10] Dorien Herremans. Morpheus: automatic music generation with recurrent pattern constraints and tension profiles. 2016.

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. Technical Report 1503.02531, arXiv, 2015.

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] Fumitada Itakura. Analysis synthesis telephony based on the maximum likelihood method. In *The 6th international congress on acoustics, 1968*, pages 280–292, 1968.

[14] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. Technical Report 1802.08435, arXiv, 2018.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[16] Neil A Macmillan and C Douglas Creelman. *Detection theory: A user's guide*. Psychology press, 2004.

[17] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. Samplernn: An unconditional end-to-end neural audio generation model. Technical Report 1612.07837, arXiv, 2016.

[18] Aaron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C Cobo, Florian Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. Technical Report 1711.10433, arXiv, 2017.

[19] Wei Ping, Kainan Peng, Andrew Gibiansky, S Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. Deep voice 3: Scaling text-to-speech with convolutional sequence learning. In *Proc. 6th International Conference on Learning Representations*, 2018.

[20] Flávio Ribeiro, Dinei Florêncio, Cha Zhang, and Michael Seltzer. Crowdmos: An approach for crowdsourcing mean opinion score studies. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 2416–2419. IEEE, 2011.

[21] Jose Sotelo, Soroush Mehri, Kundan Kumar, Joao Felipe Santos, Kyle Kastner, Aaron Courville, and Yoshua Bengio. Char2wav: End-to-end speech synthesis. 2017.

[22] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448, 2015.

[23] Yaniv Taigman, Lior Wolf, Adam Polyak, and Eliya Nachmani. Voice synthesis for in-the-wild speakers via a phonological loop. Technical Report 1707.06588, arXiv, 2017.

[24] Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. Technical Report 1609.03499, arXiv, 2016.

[25] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. Tacotron: Towards end-to-end speech synthesis. Technical Report 1703.10135, arXiv, 2017.

[26] Ronald J Williams and David Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. *Backpropagation: Theory, architectures, and applications*, 1:433–486, 1995.

Bibliography

Gammatone-based spectrograms, using gammatone filterbanks or Fourier transform weightings. URL `https://github.com/detly/gammatone`.

Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional Neural Networks for Speech Recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22:1533–1545, 2014.

Manu Airaksinen. Analysis/synthesis comparison of vocoders utilized in statistical parametric speech synthesis. *Master's thesis, Aalto University*, 2012.

Md. Jahangir Alam, Pierre Ouellet, Patrick Kenny, and Douglas D. O'Shaughnessy. Comparative evaluation of feature normalization techniques for speaker verification. In *NOLISP*, 2011.

Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, and others. Deep speech 2: End-to-end speech recognition in english and mandarin. In *arXiv preprint arXiv:1512.02595*, 2015.

Joakim Andén and Stéphane Mallat. Deep Scattering Spectrum. *IEEE Transactions on Signal Processing*, 62:4114–4128, 2014.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*, 2017.

Richard N Aslin. Some developmental processes in speech perception. *Child Phonology: Perception & Production*, 1980.

Leonardo Badino, Alessio Mereta, and Lorenzo Rosasco. Discovering discrete subword units with binarized autoencoders and hidden-markov-model encoders. In *INTERSPEECH*, 2015.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473, 2014.

James Baker. The dragon system–an overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):24–29, 1975.

Randall Balestriero, Romain Cosentino, Hervé Glotin, and Richard G. Baraniuk. Spline filters for end-to-end deep learning. In *ICML*, 2018.

Jon Barker, Emmanuel Vincent, Ning Ma, Heidi Christensen, and Phil Green. The pascal chime speech separation and recognition challenge. *Computer Speech & Language*, 27(3): 621–633, 2013.

Jon Barker, Ricard Marxer, Emmanuel Vincent, and Shinji Watanabe. The third 'chime'speech separation and recognition challenge: Dataset, task and baselines. In *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*, pages 504–511. IEEE, 2015.

Eric Battenberg, Rewon Child, Adam Coates, Christopher Fougner, Yashesh Gaur, Jiaji Huang, Heewoo Jun, Ajay Kannan, Markus Kliegl, Atul Kumar, and others. Reducing Bias in Production Speech Models. *arXiv preprint arXiv:1705.04400*, 2017.

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

MD Bedworth, L Bottou, JS Bridle, F Fallside, L Flynn, F Fogelman, KM Ponting, and RW Prager. Comparison of neural and conventional classifiers on a speech recognition problem. In *Artificial Neural Networks, 1989., First IEE International Conference on (Conf. Publ. No. 313)*, pages 86–89. IET, 1989.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. Listen and translate: A proof of concept for end-to-end speech-to-text translation. *arXiv preprint arXiv:1612.01744*, 2016.

C. Bhat, B. Vachhani, and S. K. Kopparapu. Automatic assessment of dysarthria severity level using audio descriptors. In *ICASSP*, pages 5070–5074, March 2017. doi: 10.1109/ICASSP.2017.7953122.

Léon Bottou, F Fogelman Soulié, Pascal Blanchet, and Jean-Sylvain Lienard. Experiments with time delay networks and dynamic time warping for speaker independent isolated digits recognition. In *First European Conference on Speech Communication and Technology*, 1989.

Hervé Bredin. TristouNet: Triplet Loss for Speaker Turn Embedding. *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, 2016.

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature verification using a "Siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993.

Andrew Butcher et al. *Australian Aboriginal Languages: Consonant Salient Phonologies and the'place-of-articulation Imperative'*. Australian Speech Science and Technology Association, 2003.

Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

William Chan and Ian Lane. Deep recurrent neural networks for acoustic modelling. *arXiv preprint arXiv:1504.01482*, 2015.

William Chan, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals. Listen, Attend and Spell. *CoRR*, abs/1508.01211, 2015.

Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *The Journal of Machine Learning Research*, 11: 1109–1135, 2010.

Hongjie Chen, Cheung-Chi Leung, Lei Xie, Bin Ma, and Haizhou Li. Parallel inference of dirichlet process gaussian mixture models for unsupervised acoustic modeling: a feasibility study. In *INTERSPEECH*, 2015.

Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778. IEEE, 2018.

Jan Chorowski and Navdeep Jaitly. Towards better decoding and language model integration in sequence to sequence models. *arXiv:1612.02695*, 2016.

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in neural information processing systems*, pages 577–585, 2015.

Yu-An Chung, Wei-Hung Weng, Schrasing Tong, and James Glass. Unsupervised cross-modal alignment of speech and text embedding spaces. *arXiv preprint arXiv:1805.07467*, 2018.

Renee Peje Clapham, Lisette van der Molen, R. J. J. H. van Son, Michiel W. M. van den Brekel, and Frans J. M. Hilgers. NKI-CCRT Corpus - Speech Intelligibility Before and After Advanced Head and Neck Cancer Treated with Concomitant Chemoradiotherapy. In *LREC*, 2012.

Harvey L Coates, Peter S Morris, Amanda J Leach, and Sophie Couzos. Otitis media in aboriginal children: tackling a major health problem. *The Medical Journal of Australia*, 177(4):177–178, 2002.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011.

Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve. Wav2letter: an end-to-end convnet-based speech recognition system. *arXiv preprint arXiv:1609.03193*, 2016.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *CoRR*, abs/1710.04087, 2017.

Yann Dauphin, Angela Fan, Michael Auli, and David Grangier. Language Modeling with Gated Convolutional Networks. In *ICML*, 2017.

Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366, 1980.

Alexandre Défossez, Neil Zeghidour, Nicolas Usunier, Léon Bottou, and Francis Bach. Sing: Symbol-to-instrument neural generator. In *Advances in Neural Information Processing Systems*, pages 9055–9065, 2018.

Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2011.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

Mark Dredze, Aren Jansen, Glen Coppersmith, and Ken Ward Church. Nlp on spoken documents without asr. In *EMNLP*, 2010.

Ewan Dunbar, Xuan Nga Cao, Juan Benjumea, Julien Karadayi, Mathieu Bernard, Laurent Besacier, Xavier Anguera, and Emmanuel Dupoux. The zero resource speech challenge 2017. In *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*, pages 323–330. IEEE, 2017.

Florian Eyben. *Real-time speech and music classification by large audio feature space extraction*. Springer, 2015.

Florian Eyben, Martin Wöllmer, and Björn W. Schuller. Opensmile: the munich versatile and fast open-source audio feature extractor. In *ACM Multimedia*, 2010.

Gunnar Fant. Analysis and synthesis of speech processes. *Manual of phonetics*, 2:173–277, 1968.

Gunnar Fant. *Acoustic theory of speech production: with calculations based on X-ray studies of Russian articulations*. Number 2. Walter de Gruyter, 1970.

Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.

Gustav Fechner. Elements of psychophysics. vol. i. 1966.

Naomi H Feldman, Emily B Myers, Katherine S White, Thomas L Griffiths, and James L Morgan. Word-level information influences phonetic learning in adults and infants. *Cognition*, 127(3):427–438, 2013.

Ronald Aylmer Fisher. Statistical methods for research workers. In *Statistical methods for research workers*. 1925.

JL Flanagan. Parametric coding of speech spectra. *The Journal of the Acoustical Society of America*, 68(2):412–419, 1980.

Josué Fredes, José Novoa, Simon King, Richard M. Stern, and Néstor Becerra Yoma. Locally normalized filter banks applied to deep neural-network-based robust speech recognition. *IEEE Signal Processing Letters*, 24:377–381, 2017.

Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, David S Pallett, Nancy L Dahlgren, and Victor Zue. TIMIT acoustic-phonetic continuous speech corpus. *Linguistic data consortium*, 10(5):0, 1993.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. A Convolutional Encoder Model for Neural Machine Translation. In *ACL*, 2017a.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann Dauphin. Convolutional Sequence to Sequence Learning. In *ICML*, 2017b.

Pegah Ghahremani, Vimal Manohar, Daniel Povey, and Sanjeev Khudanpur. Acoustic Modelling from the Signal Domain Using CNNs. 2016.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, pages 1764–1772, 2014.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE, 2013.

Dr. D. Greenwood. The mel scale's disqualifying bias and a consistency of pitch-difference equisections in 1956 with equal cochlear distances and equal frequency ratios. *Hearing research*, 103 1-2:199–224, 1997.

Hossein Hadian, Hossein Sameti, Daniel Povey, and Sanjeev Khudanpur. End-to-end Speech Recognition Using Lattice-free MMI. In *Interspeech*, 2018.

Kyu J Han, Akshay Chandrashekaran, Jungsuk Kim, and Ian Lane. The CAPIO 2017 Conversational Speech Recognition System. In *arXiv preprint arXiv:1801.00059*, 2017.

Awni Y. Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep Speech: Scaling up end-to-end speech recognition. *arXiv:1412.5567*, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *CVPR*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of CVPR*, 2016.

Florian Hilger and Hermann Ney. Quantile based histogram equalization for noise robust large vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 14:845–854, 2006.

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, and others. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012a.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012b.

Julia Hirschberg, Stefan Benus, Jason M Brenier, Frank Enos, Sarah Friedman, Sarah Gilman, Cynthia Girand, Martin Graciarena, Andreas Kathol, Laura Michaelis, et al. Distinguishing deceptive from non-deceptive speech. In *Ninth European Conference on Speech Communication and Technology*, 2005.

Yedid Hoshen, Ron J Weiss, and Kevin W Wilson. Speech acoustic modeling from raw multichannel waveforms. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4624–4628. IEEE, 2015.

P. Hsiao and C. Chen. Effective Attention Mechanism in Dynamic Models for Speech Emotion Recognition. In *ICASSP*, pages 2526–2530, April 2018. doi: 10.1109/ICASSP.2018.8461431.

David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

Mark Huckvale. Exploiting speech knowledge in neural nets for recognition. *Speech Communication*, 9:1–13, 1990.

Navdeep Jaitly and Geoffrey E. Hinton. Learning a better representation of speech soundwaves using restricted boltzmann machines. *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5887, 2011.

A. Jansen and B. Van Durme. Efficient spoken term discovery using randomized algorithms. In *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, pages 401–406. IEEE, 2011.

A. Jansen, E. Dupoux, S. Goldwater, M. Johnson, S. Khudanpur, K. Church, N. Feldman, H. Hermansky, F. Metze, R. Rose, and others. A summary of the 2012 JH CLSP Workshop on zero resource speech technologies and models of early language acquisition. In *Proceedings of ICASSP 2013*, 2013.

Frederick Jelinek. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556, 1976.

Ye Jia, Melvin Johnson, Wolfgang Macherey, Ron J Weiss, Yuan Cao, Chung-Cheng Chiu, Naveen Ari, Stella Laurenzo, and Yonghui Wu. Leveraging weakly supervised data to improve end-to-end speech-to-text translation. *arXiv preprint arXiv:1811.02050*, 2018.

Maree Johnson, Samuel Lapkin, Vanessa Long, Paula Sanchez, Hanna Suominen, Jim Basilakis, and Linda Dawson. A systematic review of speech recognition technology in health care. In *BMC Med. Inf. & Decision Making*, 2014.

Bing-Hwang Juang, Stephene Levinson, and M Sondhi. Maximum likelihood estimation for multivariate mixture observations of markov chains (corresp.). *IEEE Transactions on Information Theory*, 32(2):307–309, 1986.

Selen Hande Kabil, Hannah Muckenhirn, and Mathew Magimai-Doss. On learning to identify genders from raw speech signal using cnns. In *Interspeech*, 2018.

Herman Kamper, Aren Jansen, and Sharon Goldwater. Fully unsupervised small-vocabulary speech recognition using a segmental bayesian model. In *INTERSPEECH*, 2015a.

Herman Kamper, Weiran Wang, and Karen Livescu. Deep convolutional acoustic word embeddings using word-pair side information. *arXiv preprint arXiv:1510.01032*, 2015b.

Tae Gyoon Kang, Kang Hyun Lee, Woo Hyun Kang, Soo Hyun Bae, and Nam Soo Kim. Dnn-based voice activity detection with local feature shift technique. *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pages 1–4, 2016.

L G Kersta. Voiceprint identification. *Nature*, 196:1253–1257, 1962.

Heejin Kim, Mark Hasegawa-Johnson, Adrienne Perlman, Jon Gunderson, Thomas S. Huang, Kenneth Watkin, and Simone Frame. Dysarthric speech database for universal access research. In *INTERSPEECH*, 2008.

Jangwon Kim, Naveen Kumar, Andreas Tsiartas, Ming Li, and Shrikanth S. Narayanan. Automatic intelligibility classification of sentence-level pathological speech. *Computer Speech & Language*, 29(1):132 – 144, 2015. ISSN 0885-2308. doi: https://doi.org/10.1016/j.csl.2014.02.001. URL `http://www.sciencedirect.com/science/article/pii/S088523081400014X`.

Myungjong Kim, J Yoo, and H Kim. Dysarthric speech recognition using dysarthria-severity-dependent and speaker-adaptive models. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, pages 3622–3626, 2013.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 4835–4839. IEEE, 2017.

Dietrich Klakow and Jochen Peters. Testing the correlation of word error rate and perplexity. *Speech Communication*, 38(1-2):19–28, 2002.

W Koening. A new frequency scale for acoustic measurements. *Bell Lab Rec.*, pages 299–301, 1949.

Zvi Kons and Orith Toledo-Ronen. Audio event classification using deep neural networks. In *INTERSPEECH*, 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

D. S. Pavan Kumar. Feature normalisation for robust speech recognition. *CoRR*, abs/1507.04019, 2015.

Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, and Marc'Aurelio Ranzato. Fader networks: Manipulating images by sliding attributes. In *NIPS*, 2017.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. *arXiv preprint arXiv:1804.07755*, 2018.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Kai-Fu Lee and Hsiao-Wuen Hon. Speaker-independent phone recognition using hidden markov models. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 37:1641–1648, 1988.

Yun Lei, Nicolas Scheffer, Luciana Ferrer, and Mitchell McLaren. A novel scheme for speaker recognition using a phonetically-aware deep neural network. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 1695–1699. IEEE, 2014.

Stephen E Levinson, Lawrence R Rabiner, and Man Mohan Sondhi. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *Bell System Technical Journal*, 62(4):1035–1074, 1983.

Minkyu Lim, Donghyun Lee, Hosung Park, Unsang Park, and Ji-Hwan Kim. Audio event classification using deep neural networks. 2016.

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Peter H Lindsay and Donald A Norman. *Human information processing: An introduction to psychology*. Academic press, 2013.

Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert. Letter-Based Speech Recognition with Gated ConvNets. *CoRR*, abs/1712.09444, 2017. URL `http://arxiv.org/abs/1712.09444`.

Max A. Little, Patrick E. McSharry, Eric J. Hunter, Jennifer L. Spielman, and Lorraine O. Ramig. Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's Disease. *IEEE Transactions on Biomedical Engineering*, 56:1015–1022, 2009.

Chunxi Liu, Jan Trmal, Matthew Wiesner, Craig Harman, and Sanjeev Khudanpur. Topic identification for speech without asr. In *INTERSPEECH*, 2017a.

Hairong Liu, Zhenyao Zhu, Xiangang Li, and Sanjeev Satheesh. Gram-ctc: Automatic unit selection and target decomposition for sequence labelling. *arXiv preprint arXiv:1703.00096*, 2017b.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

Vincent Lostanlen, Justin Salamon, Andrew Farnsworth, Steve Kelling, and Juan Pablo Bello. Birdvox-full-night: A dataset and benchmark for avian flight call detection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 266–270. IEEE, 2018.

Vincent Lostanlen, Justin Salamon, Mark Brozier Cartwright, Brian McFee, Andrew Farnsworth, Steve Kelling, and Juan Pablo Bello. Per-channel energy normalization: Why and how. *IEEE Signal Processing Letters*, 26:39–43, 2019.

David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

Liang Lu, Lingpeng Kong, Chris Dyer, Noah A Smith, and Steve Renals. Segmental recurrent neural networks for end-to-end speech recognition. *arXiv preprint arXiv:1603.00223*, 2016.

James G. Lyons and Kuldip K. Paliwal. Effect of compressing the dynamic range of the power spectrum in modulation filtering based speech enhancement. In *INTERSPEECH*, 2008.

Ranniery Maia, Tomoki Toda, Heiga Zen, Yoshihiko Nankaku, and Keiichi Tokuda. An excitation model for hmm-based speech synthesis based on residual modeling. In *SSW*, 2007.

John Makhoul and Lynn Cosell. Lpcw: An lpc vocoder with linear predictive spectral warping. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'76.*, volume 1, pages 466–469. IEEE, 1976.

Andrew Martin, Sharon Peperkamp, and Emmanuel Dupoux. Learning phonemes with a proto-lexicon. *Cognitive science*, 37 1:103–24, 2013.

Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, pages 18–25, 2015.

Kinfe Mengistu and Frank Rudzicz. Adapting acoustic and lexical models to dysarthric speech. In *ICASSP*, pages 4924–4927, 2011a.

Kinfe Mengistu and Frank Rudzicz. Comparing Humans and Automatic Speech Recognition Systems in Recognizing Dysarthric Speech. volume 6657, pages 291–300, 2011b.

Yajie Miao, Mohammad Gowayyed, and Florian Metze. EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding. In *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2015.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH*, 2010.

J. Millet and Neil Zeghidour. Learning to detect dysarthria from raw speech. *CoRR*, abs/1811.11101, 2018.

Ji Ming and F Jack Smith. Improved phone recognition using bayesian triphone models. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1, pages 409–412. IEEE, 1998.

Abdel-rahman Mohamed, George Dahl, and Geoffrey Hinton. Deep belief networks for phone recognition. In *Nips workshop on deep learning for speech recognition and related applications*, volume 1, page 39. Vancouver, Canada, 2009.

Mehryar Mohri, Fernando Pereira, and Michael Riley. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88, 2002.

Nelson Morgan, Hervé Bourlard, and Hynek Hermansky. Automatic speech recognition: An auditory perspective. In *Speech processing in the auditory system*, pages 309–338. Springer, 2004.

Hannah Muckenhirn, Mathew Magimai-Doss, and Sébastien Marcel. Towards directly modeling raw speech signal for speaker verification using cnns. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4884–4888, 2018.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

Joy Nicholson, Kazuhiko Takahashi, and Ryohei Nakatsu. Emotion recognition in speech using neural networks. *Neural computing & applications*, 9(4):290–296, 2000.

A Michael Noll. Cepstrum pitch determination. *The journal of the acoustical society of America*, 41(2):293–309, 1967.

A Michael Noll and Manfred R Schroeder. Short-time "cepstrum" pitch detection. *The Journal of the Acoustical Society of America*, 36(5):1030–1030, 1964.

Tsubasa Ochiai, Shinji Watanabe, Takaaki Hori, and John R. Hershey. Multichannel end-to-end speech recognition. In *ICML*, 2017.

D. Kimbrough Oller, P. Niyogi, Sharmistha S. Gray, Jeffrey A. Richards, Jill Gilkerson, Daoyi Xu, Umit Yapanel, and Steven F Warren. Automated vocal analysis of naturalistic recordings from children with autism, language delay, and typical development. *Proceedings of the National Academy of Sciences of the United States of America*, 107 30:13354–9, 2010.

Mohamed Kamal Omar and Jason W. Pelecanos. A novel approach to detecting non-native speakers and their native language. *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4398–4401, 2010.

Douglas O'shaughnessy. *Speech communication: human and machine*. Universities press, 1987.

Federica Pace, Frederic Benard, Herve Glotin, Olivier Adam, and Paul White. Subunit definition and analysis for humpback whale call classification. *Applied Acoustics*, 71(11): 1107–1112, 2010.

Dimitri Palaz, Ronan Collobert, and Mathew Magimai-Doss. End-to-end phoneme sequence recognition using convolutional neural networks. *CoRR*, abs/1312.2137, 2013a.

Dimitri Palaz, Ronan Collobert, and Mathew Magimai-Doss. Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks. In *INTERSPEECH*, 2013b.

Dimitri Palaz, Mathew Magimai Doss, and Ronan Collobert. Convolutional neural networks-based continuous speech recognition using raw speech signal. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4295–4299. IEEE, 2015.

Dimitri Palaz, Gabriel Synnaeve, and Ronan Collobert. Jointly Learning to Locate and Classify Words Using Convolutional Networks. 2016.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an ASR corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE, 2015.

Alex S Park and James R Glass. Unsupervised pattern discovery in speech. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(1):186–197, 2008.

Douglas B Paul and Janet M Baker. The design for the Wall Street Journal-based CSR corpus. In *Proceedings of the workshop on Speech and Natural Language*, pages 357–362. Association for Computational Linguistics, 1992.

V. Peddinti, T. Sainath, S. Maymon, B. Ramabhadran, D. Nahamoo, and V. Goel. Deep scattering spectrum with deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 210–214. IEEE, 2014.

Matthew Perez, Wenyu Jin, Duc Le, Noelle Carlozzi, Praveen Dayalu, Angela Roberts, and Emily Mower Provost. Classification of huntington disease using acoustic and lexical features. In *Interspeech*, 2018.

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer, 2010.

Gueorgui Pironkov, Stéphane Dupont, and Thierry Dutoit. Speaker-aware long short-term memory multi-task learning for speech recognition. In *Signal Processing Conference (EUSIPCO), 2016 24th European*, pages 1911–1915. IEEE, 2016.

M.A. Pitt, L. Dilley, K. Johnson, S. Kiesling, W. Raymond, E. Hume, and E. Fosler-Lussier. *Buckeye Corpus of Conversational Speech (2nd release)*. Columbus, OH: Department of Psychology, Ohio State University (Distributor), 2007. Published: www.buckeyecorpus.osu.edu.

Heather Pon-Barry. Prosodic manifestations of confidence and uncertainty in spoken language. In *INTERSPEECH*, 2008.

Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohammadi, and Sanjeev Khudanpur. Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Interspeech*, 2018.

Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Kanishka Rao, Haşim Sak, and Rohit Prabhavalkar. Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 193–199. IEEE, 2017.

Mirco Ravanelli and Yoshua Bengio. Speaker recognition from raw waveform with sincnet. *arXiv preprint arXiv:1808.00158*, 2018.

Steve Renals, Nelson Morgan, Hervé Bourlard, Michael Cohen, and Horacio Franco. Connectionist probability estimators in hmm speech recognition. *IEEE Trans. Speech and Audio Processing*, 2:161–174, 1994.

Daniel Renshaw, Herman Kamper, Aren Jansen, and Sharon Goldwater. A comparison of neural network methods for unsupervised representation learning on the zero resource speech challenge. In *INTERSPEECH*, 2015.

Frank Rudzicz, Pascal Van Lieshout, Graeme Hirst, Gerald Penn, Fraser Shein, and Talya Wolff. Towards a Comparative Database of Dysarthric Articulation. In *Proceedings of ISSP*, 2008.

Frank Rudzicz, Aravind Kumar Namasivayam, and Talya Wolff. The TORGO database of acoustic and articulatory speech from speakers with dysarthria. *Language Resources and Evaluation*, 46(4):523–541, December 2012. ISSN 1574-0218. doi: 10.1007/s10579-011-9145-0. URL `https://doi.org/10.1007/s10579-011-9145-0`.

Seyed Omid Sadjadi, Sriram Ganapathy, and Jason W Pelecanos. The IBM 2016 speaker recognition system. *arXiv preprint arXiv:1602.07291*, 2016.

Tara N Sainath, Brian Kingsbury, Abdel-rahman Mohamed, and Bhuvana Ramabhadran. Learning filter banks within a deep neural network framework. In *ASRU*, pages 297–302. IEEE, 2013.

Tara N Sainath, Ron J Weiss, Andrew Senior, Kevin W Wilson, and Oriol Vinyals. Learning the speech front-end with raw waveform CLDNNs. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015a.

Tara N. Sainath, Ron J. Weiss, Kevin W. Wilson, Arun Narayanan, Michiel Bacchiani, and Andrew W. Senior. Speaker location and microphone spacing invariant acoustic modeling from raw multichannel waveforms. *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 30–36, 2015b.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

Justin Salamon, Juan Pablo Bello, Andrew Farnsworth, and Steve Kelling. Fusing shallow and deep learning for bioacoustic bird species classification. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 141–145, 2017.

Tim Salimans and Diederik P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909, 2016.

George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, and others. English conversational telephone speech recognition by humans and machines. *arXiv preprint arXiv:1703.02136*, 2017.

Shimon Sapir, Lorraine O. Ramig, Jennifer L. Spielman, and Cynthia Fox. Formant centralization ratio: a proposal for a new acoustic measure of dysarthric speech. *Journal of speech, language, and hearing research : JSLHR*, 53 1:114–25, 2010.

Mousmita Sarma, Pegah Ghahremani, Daniel Povey, Nagendra Kumar Goel, Kandarpa Kumar Sarma, and Najim Dehak. Emotion identification from raw speech signals using dnns. In *Interspeech*, 2018.

T. Schatz, V. Peddinti, F. Bach, A. Jansen, H. Hermansky, and E. Dupoux. Evaluating speech features with the minimal-pair abx task: Analysis of the classical mfc/plp pipeline. In *INTERSPEECH 2013: 14th Annual Conference of the International Speech Communication Association*, pages 1–5, 2013.

Thomas Schatz. *ABX-discriminability measures and applications.* PhD thesis, Université Paris 6 (UPMC), 2016.

Ralf Schlüter, Ilja Bezrukov, Hermann Wagner, and Hermann Ney. Gammatone features and feature combination for large vocabulary speech recognition. *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, 4:IV–649–IV–652, 2007.

Björn Schuller, Stefan Steidl, Anton Batliner, Felix Burkhardt, Laurence Devillers, Christian MüLler, and Shrikanth Narayanan. Paralinguistics in speech and language—state-of-the-art and the challenge. *Computer Speech & Language*, 27(1):4–39, 2013a.

Björn Schuller, Stefan Steidl, Anton Batliner, Alessandro Vinciarelli, Klaus Scherer, Fabien Ringeval, Mohamed Chetouani, Felix Weninger, Florian Eyben, Erik Marchi, et al. The interspeech 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism. In *Proceedings INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France*, 2013b.

Björn W. Schuller, Stefan Steidl, Anton Batliner, Felix Burkhardt, Laurence Devillers, Christian A. Müller, and Shrikanth Narayanan. The interspeech 2010 paralinguistic challenge. In *INTERSPEECH*, 2010.

Björn W. Schuller, Stefan Steidl, Anton Batliner, Elika Bergelson, Jarek Krajewski, Christoph Janott, Andrei Amatuni, Marisa Casillas, Amanda Seidl, Melanie Soderstrom, Anne S. Warlaumont, Guillermo Hidalgo, Sebastian Schnieder, Clemens Heiser, Winfried Hohenhorst, Michael Herzog, Maximilian Schmitt, Kun Qian, Yue Zhang, George Trigeorgis, Panagiotis Tzirakis, and Stefanos P. Zafeiriou. The interspeech 2017 computational paralinguistics challenge: Addressee, cold & snoring. In *INTERSPEECH*, 2017.

Björn W. Schuller, Yue Zhang, and Felix Weninger. Three recent trends in paralinguistics on the way to omniscient machine intelligence. *Journal on Multimodal User Interfaces*, 12:273–283, 2018.

Björn Schuller, Stefan Steidl, and Anton Batliner. The Interspeech 2009 Emotion Challenge. In *Proc. Interspeech*, pages 312–315, 2009.

Hiroshi Seki, Takaaki Hori, Shinji Watanabe, Jonathan Le Roux, and John R Hershey. A purely end-to-end system for multi-speaker speech recognition. *arXiv preprint arXiv:1805.05826*, 2018.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Malcolm Slaney. Auditory toolbox. *Interval Research Corporation, Tech. Rep*, 10:1998, 1998.

Evan C Smith and Michael S Lewicki. Efficient auditory coding. *Nature*, 439(7079):978–982, 2006.

S Smith Stevens. On the psychophysical law. *Psychological review*, 64 3:153–81, 1957.

Stanley S Stevens and John Volkmann. The relation of pitch to frequency: A revised scale. *The American Journal of Psychology*, 53(3):329–353, 1940.

Stanley Smith Stevens, John Volkmann, and Edwin B Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Daniel Swingley. Contributions of infant word learning to language development. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 364 1536: 3617–32, 2009.

Gabriel Synnaeve and Emmanuel Dupoux. Weakly Supervised Multi-Embeddings Learning of Acoustic Models. In *ICLR*, 2014.

Gabriel Synnaeve, Thomas Schatz, and Emmanuel Dupoux. Phonetics Embedding Learning with Side Information. In *IEEE Spoken Language Technology Workshop*. IEEE, 2014.

Zhiyuan Tang, Lantian Li, and Dong Wang. Multi-task recurrent model for speech and speaker recognition. *arXiv preprint arXiv:1603.09643*, 2016.

Joseph Tepperman, David Traum, and Shrikanth Narayanan. " yeah right": Sarcasm recognition for spoken dialogue systems. In *Ninth International Conference on Spoken Language Processing*, 2006.

R. Thiollière, E. Dunbar, G. Synnaeve, M. Versteegh, and E. Dupoux. A Hybrid Dynamic Time Warping-Deep Neural Network Architecture for Unsupervised Acoustic Modeling. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Attention-based Wav2text with Feature Transfer Learning. *arXiv preprint arXiv:1709.07814*, 2017a.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Sequence-to-Sequence ASR Optimization via Reinforcement Learning. *arXiv preprint arXiv:1710.10774*, 2017b.

Shubham Toshniwal, Tara N Sainath, Ron J Weiss, Bo Li, Pedro Moreno, Eugene Weinstein, and Kanishka Rao. Multilingual speech recognition with a single end-to-end model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4904–4908. IEEE, 2018.

Oscar Tosi, Herbert Oyer, William Lashbrook, Charles Pedrey, Julie Nicol, and Ernest Nash. Experiment on voice identification. *The Journal of the Acoustical Society of America*, 51 (6B):2030–2043, 1972.

László Tóth. Combining time-and frequency-domain convolution in convolutional neural network-based phone recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 190–194. IEEE, 2014.

George Trigeorgis, Fabien Ringeval, Raymond Brueckner, Erik Marchi, Mihalis A. Nicolaou, Björn W. Schuller, and Stefanos Zafeiriou. Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network. *ICASSP*, pages 5200–5204, 2016.

László Tóth. Phone recognition with hierarchical convolutional deep maxout networks. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015(1):25, 2015.

Zoltán Tüske, Pavel Golik, Ralf Schlüter, and Hermann Ney. Acoustic modeling with deep neural networks using raw time signal for LVCSR. In *Interspeech*, 2014.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance Normalization: The Missing Ingredient for Fast Stylization. *arXiv preprint arXiv:1607.08022*, 2016.

Srinivasan Umesh, Leon Cohen, and D Nelson. Fitting the mel scale. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 1, pages 217–220. IEEE, 1999.

Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Advances in neural information processing systems*, pages 2643–2651, 2013.

Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. Technical Report 1609.03499, arXiv, 2016.

Laurens Van Der Maaten and Kilian Weinberger. Stochastic triplet embedding. In *Machine Learning for Signal Processing (MLSP), 2012 IEEE International Workshop on*, pages 1–6. IEEE, 2012.

Tom Véniat, Olivier Schwander, and Ludovic Denoyer. Stochastic adaptive neural architecture search for keyword spotting. *arXiv preprint arXiv:1811.06753*, 2018.

Maarten Versteegh, Roland Thiolliere, Thomas Schatz, Xuan Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. The zero resource speech challenge 2015. In *Proc. of Interspeech*, 2015.

Olli Viikki and Kari Laurila. Cepstral domain segmental feature vector normalization for noise robust speech recognition. *Speech Communication*, 25(1-3):133–147, 1998.

Olli Viikki, David Bye, and Kari Laurila. A recursive feature vector normalization approach for robust speech recognition in noise. In *ICASSP*, 1998.

Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269, 1967.

N. J. de Vries, M. H. Davel, J. Badenhorst, W. D. Basson, F. de Wet, E. Barnard, and A. de Waal. A smartphone-based ASR data collection tool for under-resourced languages. *Speech Communication*, 56:119–131, 2014.

Alexander H. Waibel, Toshiyuki Hanazawa, Geoffrey E. Hinton, Kiyohiro Shikano, and Kevin J. Lang. Phoneme recognition using time-delay neural networks. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 37:328–339, 1989.

Yu-Hsuan Wang, Hung yi Lee, and Lin-Shan Lee. Segmental audio word2vec: Representing utterances as sequences of vectors with applications in spoken term detection. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6269–6273, 2018.

Yuxuan Wang, Pascal Getreuer, Thad Hughes, Richard F Lyon, and Rif A Saurous. Trainable frontend for robust and far-field keyword spotting. In *ICASSP*, pages 5670–5674. IEEE, 2017.

Benjamin Weiss and Felix Burkhardt. Voice attributes affecting likability perception. In *INTERSPEECH*, 2010.

Yin Xian, Andrew Thompson, Qiang Qiu, Loren Nolte, Douglas Nowacek, Jianfeng Lu, and Robert Calderbank. Classification of whale vocalizations using the weyl transform. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 773–777. IEEE, 2015.

Bing Xiang, Upendra V. Chaudhari, Jirí Navrátil, Ganesh N. Ramaswamy, and Ramesh A. Gopinath. Short-time gaussianization for robust speaker verification. *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1:I–681–I–684, 2002.

Lingxi Xie and Alan Yuille. Genetic cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1388–1397. IEEE, 2017.

Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. Achieving human parity in conversational speech recognition. *arXiv preprint arXiv:1610.05256*, 2016.

Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

Takayoshi Yoshimura, Keiichi Tokuda, Takashi Masuko, Takao Kobayashi, and Tadashi Kitamura. Mixed excitation for hmm-based speech synthesis. In *INTERSPEECH*, 2001.

Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, et al. The htk book. *Cambridge university engineering department*, 3:175, 2002.

Neil Zeghidour, Gabriel Synnaeve, Nicolas Usunier, and Emmanuel Dupoux. Joint learning of speaker and phonetic similarities with siamese networks. In *INTERSPEECH*, 2016a.

Neil Zeghidour, Gabriel Synnaeve, Maarten Versteegh, and Emmanuel Dupoux. A Deep Scattering Spectrum-Deep Siamese Network Pipeline for Unsupervised Acoustic Modeling. In *ICASSP*, 2016b.

Neil Zeghidour, Nicolas Usunier, Iasonas Kokkinos, Thomas Schatz, Gabriel Synnaeve, and Emmanuel Dupoux. Learning Filterbanks from Raw Speech for Phone Recognition. *arXiv preprint arXiv:1711.01161*, 2017.

Neil Zeghidour, Nicolas Usunier, Gabriel Synnaeve, Ronan Collobert, and Emmanuel Dupoux. End-to-End Speech Recognition from the Raw Waveform. In *Interspeech*, 2018a.

Neil Zeghidour, Qiantong Xu, Vitaliy Liptchinsky, Nicolas Usunier, Gabriel Synnaeve, and Ronan Collobert. Fully convolutional speech recognition. *arXiv preprint arXiv:1812.06864*, 2018b.

Matthew D Zeiler. ADADELTA: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*. Springer, 2014.

Heiga Zen, Tomoki Toda, Masaru Nakamura, and Keiichi Tokuda. Details of the nitech hmm-based speech synthesis system for the blizzard challenge 2005. *IEICE Transactions*, 90-D:325–333, 2007.

Albert Zeyer, Kazuki Irie, Ralf Schlüter, and Hermann Ney. Improved training of end-to-end attention models for speech recognition. *arXiv preprint arXiv:1805.03294*, 2018.

Ying Zhang, Mohammad Pezeshki, Philémon Brakel, Saizheng Zhang, Cesar Laurent Yoshua Bengio, and Aaron Courville. Towards end-to-end speech recognition with deep convolutional neural networks. In *arXiv preprint arXiv:1701.02720*, 2017.

Zixing Zhang, Jing Han, Kun Qian, and Björn W. Schuller. Evolving learning for analysing mood-related infant vocalisation. In *Interspeech*, 2018.

Yingbo Zhou, Caiming Xiong, and Richard Socher. Improving End-to-End Speech Recognition with Policy Learning. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

Résumé

Bien que les réseaux de neurones soient à présent utilisés dans la quasi-totalité des composants d'un système de reconnaissance de la parole, du modèle acoustique au modèle de langue, l'entrée de ces systèmes reste une représentation analytique et fixée de la parole dans le domaine temps-fréquence, telle que les mel-filterbanks. Cela se distingue de la vision par ordinateur, un domaine où les réseaux de neurones prennent en entrée les pixels bruts. Les mel-filterbanks sont le produit d'une connaissance précieuse et documentée du système auditif humain, ainsi que du traitement du signal, et sont utilisées dans les systèmes de reconnaissance de la parole les plus en pointe, systèmes qui rivalisent désormais avec les humains dans certaines conditions. Cependant, les mel-filterbanks, comme toute représentation fixée, sont fondamentalement limitées par le fait qu'elles ne soient pas affinées par apprentissage pour la tâche considérée. Nous formulons l'hypothèse qu'apprendre ces représentations de bas niveau de la parole, coinjontement avec le modèle, permettrait de faire avancer davantage l'état de l'art. Nous explorons tout d'abord des approches d'apprentissage faiblement supervisé et montrons que nous pouvons entraîner un unique réseau de neurones à séparer l'information phonétique de celle du locuteur à partir de descripteurs spectraux ou du signal brut et que ces représentations se transfèrent à travers les langues. De plus, apprendre à partir du signal brut produit des représentations du locuteur significativement meilleures que celles d'un modèle entraîné sur des mel-filterbanks. Ces résultats encourageants nous mènent par la suite à développer une alternative aux mel-filterbanks qui peut être entraînée à partir des données. Dans la seconde partie de cette thèse, nous proposons les Time-Domain filterbanks, une architecture neuronale légère prenant en entrée la forme d'onde, dont on peut initialiser les poids pour répliquer les mel-filterbanks et qui peut, par la suite, être entraînée par rétro-propagation avec le reste du réseau de neurones. Au cours d'expériences systématiques et approfondies, nous montrons que les Time-Domain filterbanks surclassent systématiquement les mel-filterbanks, et peuvent être intégrées dans le premier système de reconnaissance de la parole purement convolutif et entraîné à partir du signal brut, qui constitue actuellement un nouvel état de l'art. Les descripteurs fixes étant également utilisés pour des tâches de classification non-linguistique, pour lesquelles elles sont d'autant moins optimales, nous entraînons un système de détection de dysarthrie à partir du signal brut, qui surclasse significativement un système équivalent entraîné sur des mel-filterbanks ou sur des descripteurs de bas niveau. Enfin, nous concluons cette thèse en expliquant en quoi nos contributions s'inscrivent dans une transition plus large vers des systèmes de compréhension du son qui pourront être appris de bout en bout.

Mots Clés

reconnaissance de la parole, signal audio, apprentissage profond, réseau de neurones

Abstract

While deep neural networks are now used in almost every component of a speech recognition system, from acoustic to language modeling, the input to such systems are still fixed, handcrafted, spectral features such as mel-filterbanks. This contrasts with computer vision, in which a deep neural network is now trained on raw pixels. Mel-filterbanks contain valuable and documented prior knowledge from human auditory perception as well as signal processing, and are the input to state-of-the-art speech recognition systems that are now on par with human performance in certain conditions. However, mel-filterbanks, as any fixed representation, are inherently limited by the fact that they are not fine-tuned for the task at hand. We hypothesize that learning the low-level representation of speech with the rest of the model, rather than using fixed features, could push the state-of-the art even further. We first explore a weakly-supervised setting and show that a single neural network can learn to separate phonetic information and speaker identity from mel-filterbanks or the raw waveform, and that these representations are robust across languages. Moreover, learning from the raw waveform provides significantly better speaker embeddings than learning from mel-filterbanks. These encouraging results lead us to develop a learnable alternative to mel-filterbanks, that can be directly used in replacement of these features. In the second part of this thesis we introduce Time-Domain filterbanks, a lightweight neural network that takes the waveform as input, can be initialized as an approximation of mel-filterbanks, and then learned with the rest of the neural architecture. Across extensive and systematic experiments, we show that Time-Domain filterbanks consistently outperform mel-filterbanks and can be integrated into a new state-of-the-art speech recognition system, trained directly from the raw audio signal. Fixed speech features being also used for non-linguistic classification tasks for which they are even less optimal, we perform dysarthria detection from the waveform with Time-Domain filterbanks and show that it significantly improves over mel-filterbanks or low-level descriptors. Finally, we discuss how our contributions fall within a broader shift towards fully learnable audio understanding systems.

Keywords

speech recognition, audio signal processing, deep learning, neural network