Raw Waveform Modelling for ASR

A Literature Review

Part IV: Parametric CNNs

Erfan Loweimi

Centre for Speech Technology Research (CSTR)
The University of Edinburgh

Listen! 14.4.2020

Outline

- Time-Frequency Analysis (TFA) without Fourier
- Parametric Kernelised CNNs
  - SincNet, Sinc^{2}Net, GammaNet, GaussNet, Complex Gabor CNN
- E2E Raw waveform models for ASR
  - Time-Domain Filterbank
  - E2E-SincNet
- Adaptation of SincNet acoustic models

TFA by Time-domain Processing

- Requires impulse response, *h(t)*, of fbank filters
  - Known for Gammatone filters

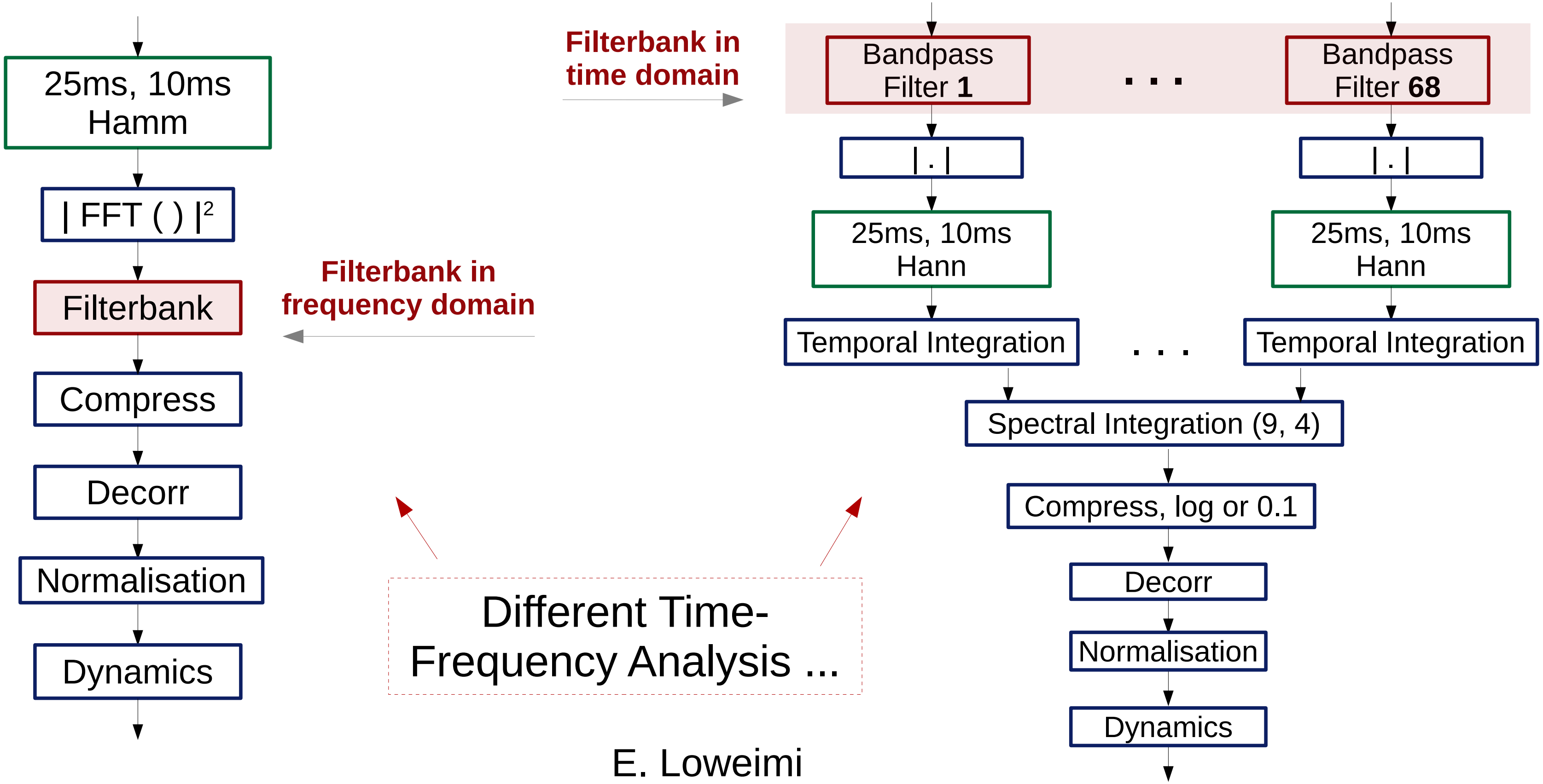
GAMMATONE FEATURES AND FEATURE COMBINATION FOR LARGE VOCABULARY SPEECH RECOGNITION

ICASSP 2007

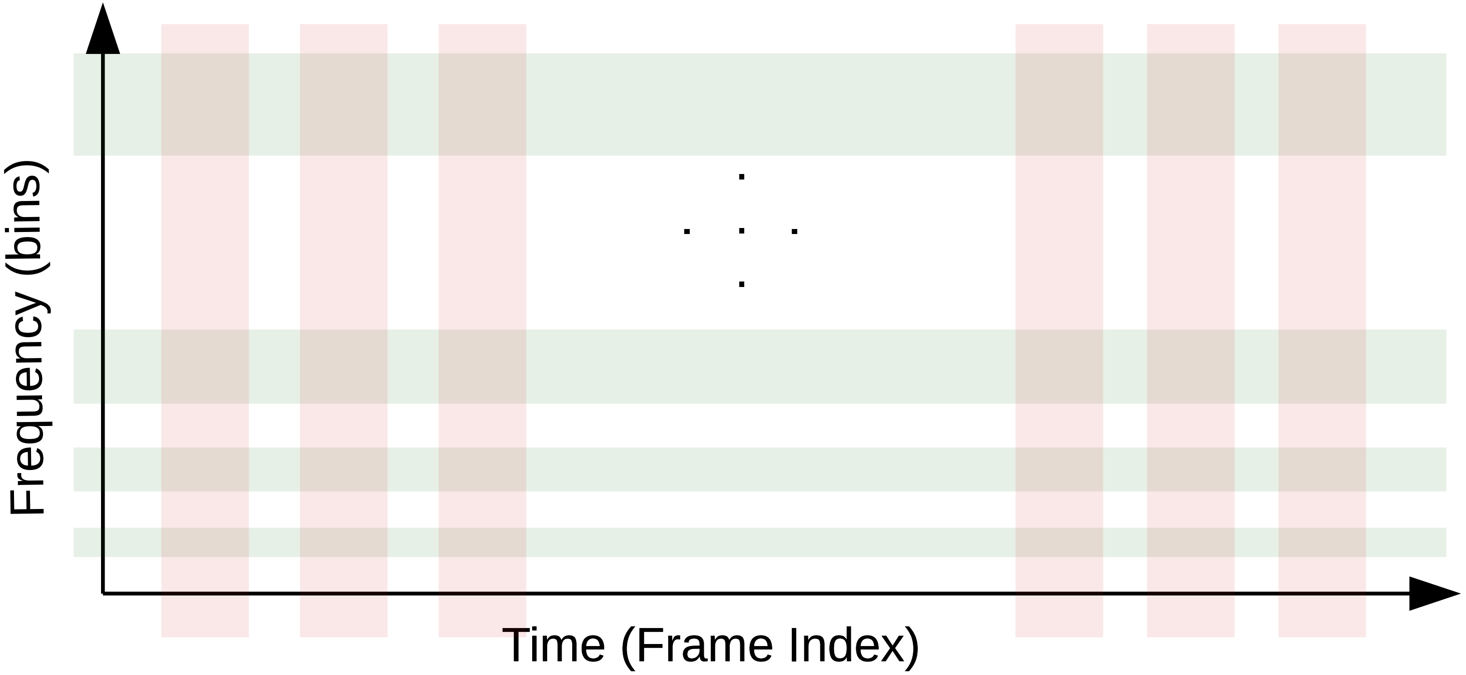
R. Schlüter[1], *I. Bezrukov*[1], *H. Wagner*[2], *H. Ney*[1]

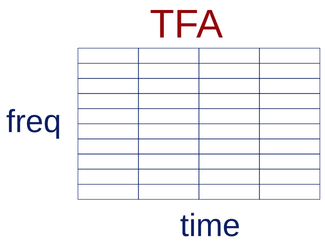
[1]Lehrstuhl für Informatik 6 - Computer Science Department
[2]Lehrstuhl für Biologie II - Biology Department
RWTH Aachen University, Aachen, Germany
schlueter@cs.rwth-aachen.de

MFCC vs Gammatone Feature

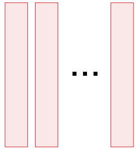


TFA in **MFCC** & **Gammatone** Pipelines

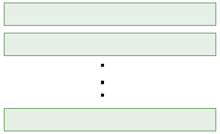




Frequency domain → hstack



Time-domain → vstack



SincNet

SPEAKER RECOGNITION FROM RAW WAVEFORM WITH SINCNET

SLT 2018

*Mirco Ravanelli, Yoshua Bengio**

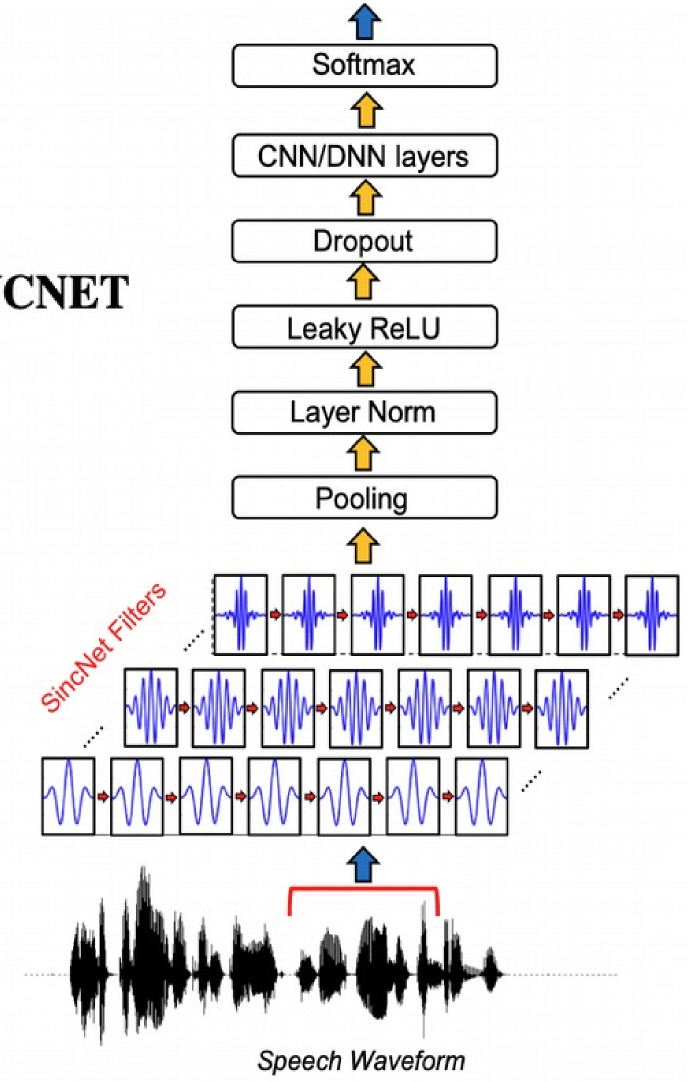
Mila, Université de Montréal, *CIFAR Fellow

Interpretable Convolutional Filters with SincNet

NIPS@IRASL 2018

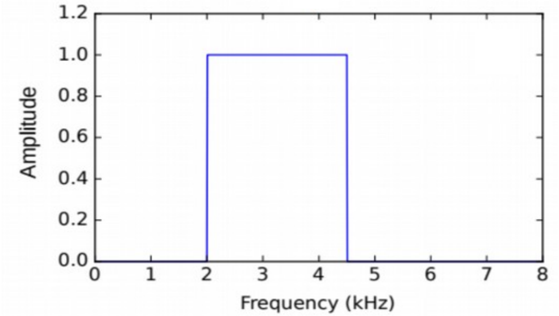
Mirco Ravanelli
Mila, Université de Montréal

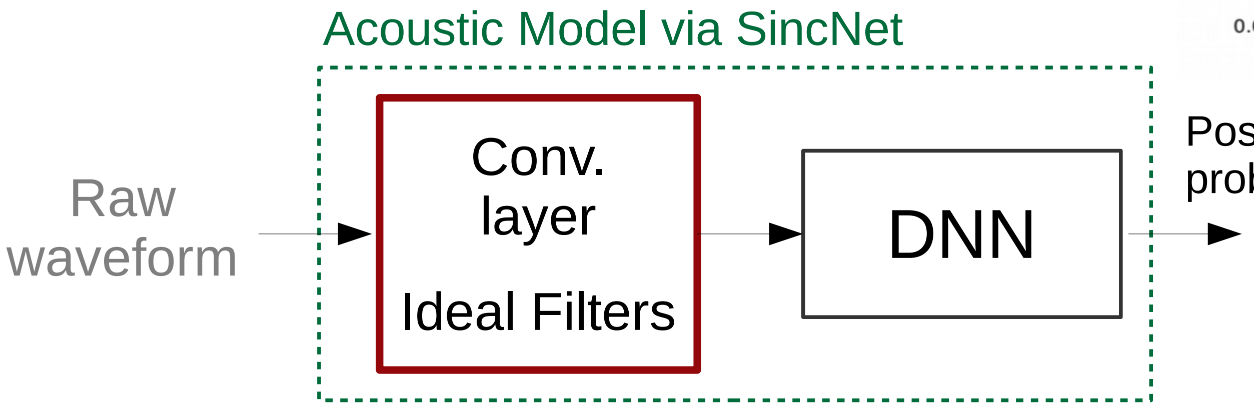
Yoshua Bengio
Mila, Université de Montréal
CIFAR Fellow



SincNet – Definition

- Convolutional layer with ideal bandpass filters, takes raw waveform as input
  - Impulse response ← Sinc





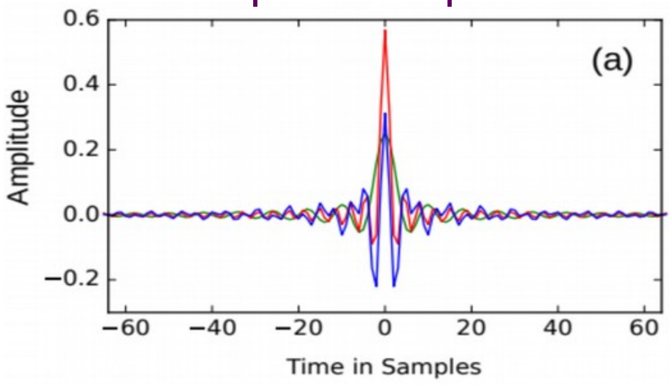
SincNet – Filters

$$sinc(x) = \frac{\sin(\pi x)}{\pi x}$$

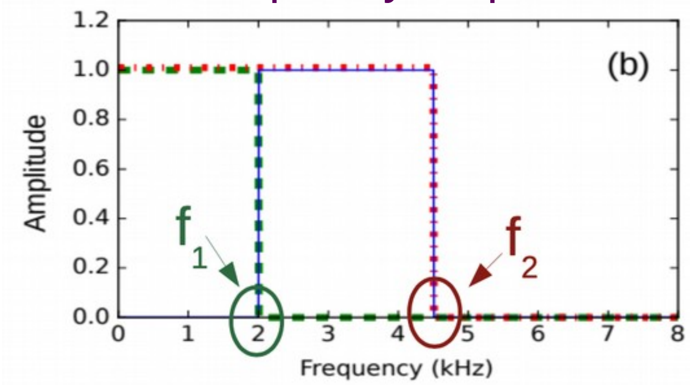
$$h(t;\theta^{(i)}) = 2f_2^{(i)} sinc(2f_2^{(i)}t) - 2f_1^{(i)} sinc(2f_1^{(i)}t)$$

$$H(f;\theta^{(i)}) = \Pi(\frac{f}{2f_2^{(i)}}) - \Pi(\frac{f}{2f_1^{(i)}})$$

Impulse response



Frequency response



SincNet – Parameters

- Parameter Set (Θ) → cut-off frequencies: f_1 & f_2

$$h(t;\theta^{(i)}) = 2f_2^{(i)} sinc(2f_2^{(i)}t) - 2f_1^{(i)} sinc(2f_1^{(i)}t)$$

$$H(f;\theta^{(i)}) = \Pi(\frac{f}{2f_2^{(i)}}) - \Pi(\frac{f}{2f_1^{(i)}})$$

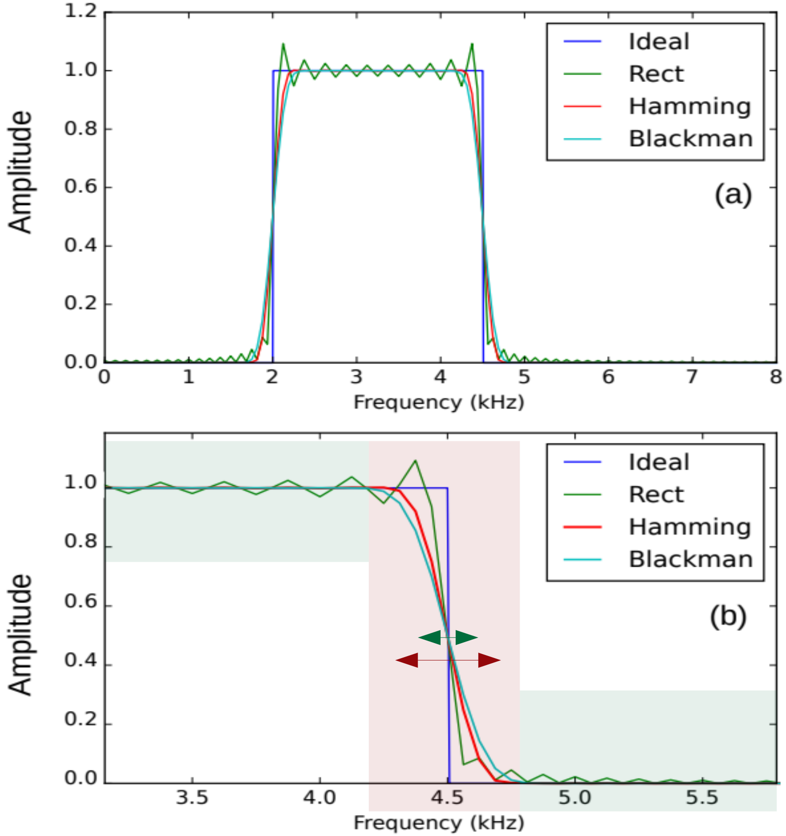
$$\Theta = \{\theta^{(i)}\} = \{f_1^{(i)}, f_2^{(i)}\}$$

Learned via Backprop

SincNet – Practical Considerations (1)

- Sinc length is finite
  - Rectangular windowing
    - Ripples in pass/stop bands
  - **Solution**:
    - Apply a tapered window

$$h(t;\theta^{(i)}) \leftarrow h(t;\theta^{(i)})\ window(t)$$



SincNet – Practical Considerations (2)

- Monitor the cut-off frequencies value
  - Both should be positive and $f_2 > f_1$
  - $f_2 <$ Nyquist Rate

$$f_1 \leftarrow |f_1|$$

$$f_2 \leftarrow f_1 + |f_2 - f_1|$$

SincNet – Practical Considerations

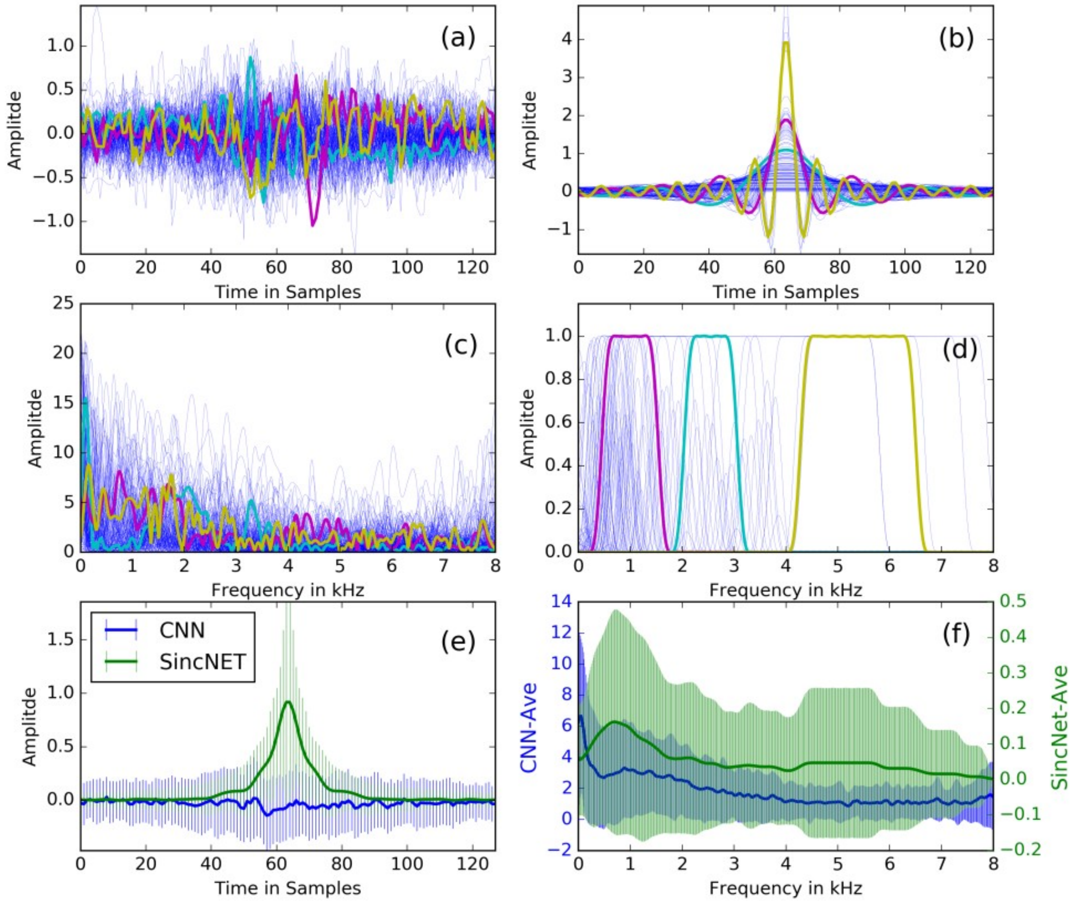
- Sinc length is finite → Apply a tapered window
- Monitor the cut-off frequencies value
- Amplitude learning is not necessary
  - Weights of the higher layer
- Initialisation of Parameters (cut-off frequencies)
  - Perceptual scale (e.g. Mel) or random initialisation

CNN vs SincNet

CNN
impulse responses

CNN
Frequency responses

Average
impulse responses



SincNet
impulse responses

SincNet
Frequency responses

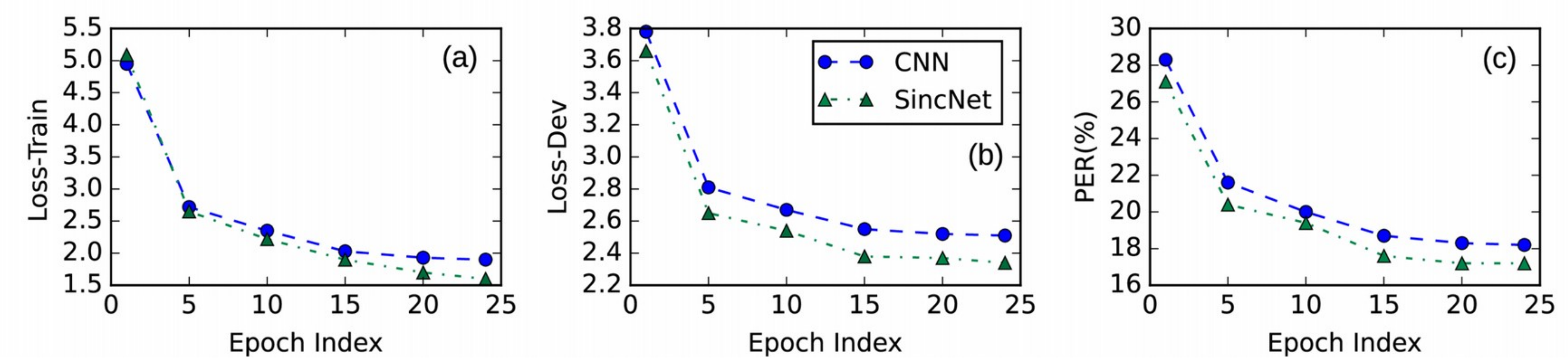
Average
Frequency responses

SincNet vs CNN -- Advantages

- **Parametric** vs Non-parametric
  - More interpretable
  - Constraint on hypothesis space
    - Regularisation → better generalisation
  - Fewer parameters
    - Less training data required
    - Faster learning/convergence

SincNet vs CNN -- Advantages

- Parametric vs Non-parametric
- Better performance on TIMIT & WSJ ...
  - Lower loss and phone error rate (PER)



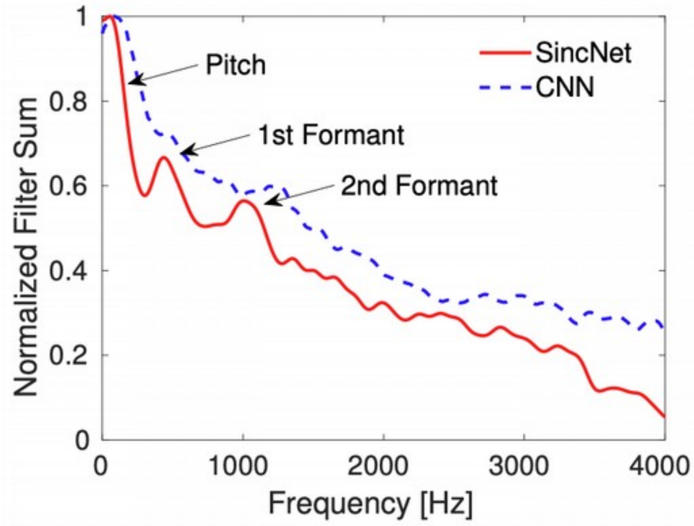
Speaker Recognition with SincNet

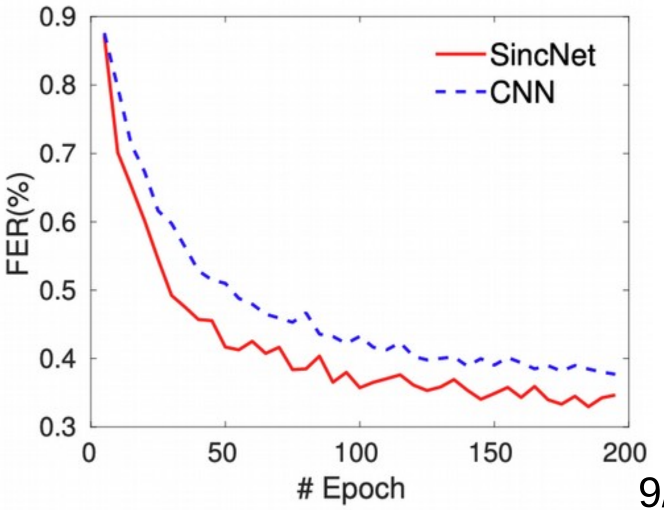
Speaker Identification Task (CER%)

	TIMIT	LibriSpeech
DNN-MFCC	0.99	2.02
CNN-FBANK	0.86	1.55
CNN-Raw	1.65	1.00
SINCNET	**0.85**	**0.96**

Speaker Verification Task (EER%)

	TIMIT	LibriSpeech
DNN-MFCC	0.99	2.02
CNN-FBANK	0.86	1.55
CNN-Raw	1.65	1.00
SINCNET	**0.85**	**0.96**





Kernelised CNNs IDEA

On Learning Interpretable CNNs with Parametric Modulated Kernel-based Filters

Erfan Loweimi, Peter Bell and Steve Renals

Centre for Speech Technology Research (CSTR), School of Informatics, University of Edinburgh

{e.loweimi, peter.bell, s.renals}@ed.ac.uk

Interpretable Kernel-based CNNs

$$h(t;\theta^{(i)}) = 2f_2^{(i)} sinc(2f_2^{(i)}t) - 2f_1^{(i)} sinc(2f_1^{(i)}t)$$

$$h(t;\theta^{(i)}) = \frac{1}{\pi t}(\sin(2\pi f_2^{(i)}t) - \sin(2\pi f_1^{(i)}t))$$

$$\sin\alpha - \sin\beta = 2\sin\frac{\alpha-\beta}{2}\cos\frac{\alpha+\beta}{2}$$

$$h^{(i)}(t) = 2B^{(i)} sinc(B^{(i)}t)\ \cos(2\pi f_c^{(i)}t)$$

$$B^{(i)} = f_2^{(i)} - f_1^{(i)} \quad , \quad f_c^{(i)} = \frac{f_1^{(i)} + f_2^{(i)}}{2}$$

Kernelised CNNs IDEA

$$h^{(i)}(t) = \boxed{2B^{(i)} sinc(B^{(i)} t)} \; \boxed{\cos(2\pi f_c^{(i)} t)}$$

Baseband filter ≡ Kernel

Carrier

$$h^{(i)}(t; \theta^{(i)}, f_c^{(i)}) = \boxed{K(t; \theta^{(i)})} \; \boxed{carrier(t; f_c^{(i)})}$$

$$\text{Parameter Set: } \Theta = \{\theta^{(i)}, f_c^{(i)}\}$$

Learned via Backprop

Kernelised CNNs IDEA

$$h^{(i)}(t;\theta^{(i)}, f_c^{(i)}) = \boxed{K(t;\theta^{(i)})} \ \boxed{carrier(t; f_c^{(i)})}$$

$$\text{Parameter Set: } \Theta = \{\theta^{(i)}, f_c^{(i)}\}$$

- **Kernel** (baseband filter) Examples
  - ✔ Sinc² → Triangular filters (similar to MFCC) → Sinc²Net
  - ✔ Gammatone → Mimics filtering in Cochlea → GammaNet
  - ✔ Gaussian → Gaussian or Gabor filter → GaussNet
  - ✔ ...

CGCNN: COMPLEX GABOR CONVOLUTIONAL NEURAL NETWORK ON RAW SPEECH

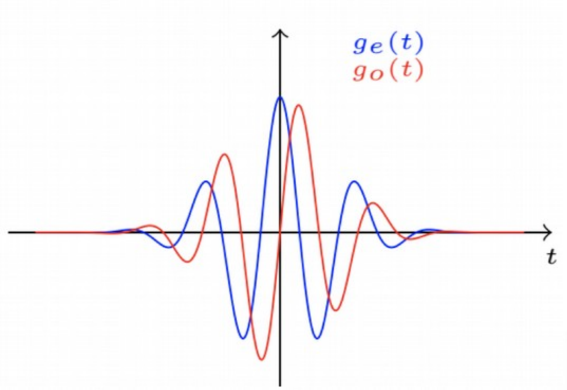
Paul-Gauthier Noé[1], *Titouan Parcollet*[1,2], *Mohamed Morchid*[1]

[1]LIA, Université d'Avignon, France
[2]University of Oxford, UK

Complex Gabor CNN (CGCNN)

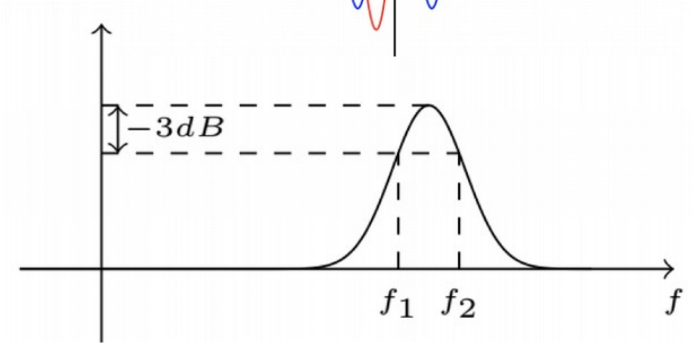
- CNN with Complex Gabor kernel

$$g(t) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{t^2}{2\sigma^2}) \exp(j2\pi f_c t)$$
$$= g_e(t) + jg_o(t)$$



$$G(f) = \exp(-2\pi^2\sigma^2(f - f_0)^2)$$

$$\sigma = \frac{A}{\pi(f_2 - f_1)} \qquad f_c = \frac{f_1 + f_2}{2}$$

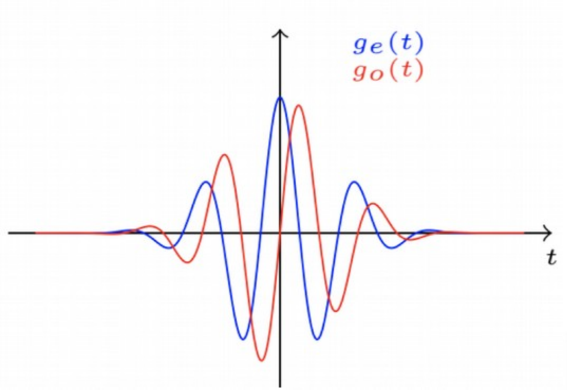


$$A = \sqrt{\frac{3ln10}{10}}$$

Complex Gabor CNN (CGCNN)

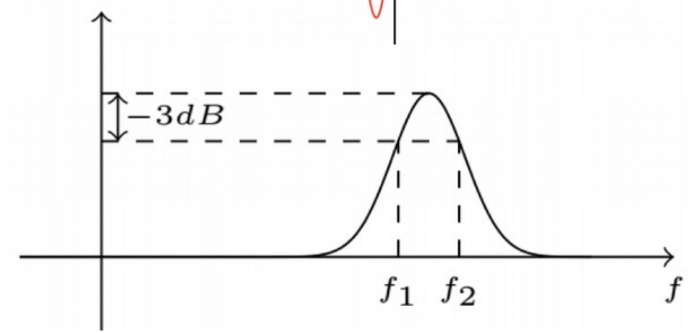
- CNN with Complex Gabor kernel

$$g(t) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{t^2}{2\sigma^2}) \exp(j2\pi f_c t)$$
$$= g_e(t) + j g_o(t)$$



$$G(f) = \exp(-2\pi^2\sigma^2(f - f_0)^2)$$

$$\sigma = \frac{A}{\pi(f_2 - f_1)} \qquad f_c = \frac{f_1 + f_2}{2}$$



$$A = \sqrt{\frac{3ln10}{10}}$$

I think *A* should be \sqrt(log2)!
Set G(f) = 1/sqrt(2) and solve for f

CGCNN Advantages / Performance

- Optimal time-frequency resolution trade-off
  - Gaussian → $\Delta t\ \Delta\omega = 0.5$; For others → $\Delta t\ \Delta\omega \geq 0.5$
- Performance is similar to GaussNet on Average
  - Best results is not reliable; How many runs?
    - Once I got 16.6% for SincNet while on ave PER is around 17.4%
  - Freq response of both Real and Complex is identical

Model	Valid.%	Avg. Test%	Best Test%
Gabor-CNN-CTC [18]	-	18.8	18.5
SincNet [2]	-	17.2	-
GaborReal	15.2	17.2	16.9
GaborComplex	15.2	17.1	16.7

CGCNN Advantages / Performance

- Optimal time-frequency resolution trade-off
  - Gaussian → $\Delta t\, \Delta\omega = 0.5$; For others → $\Delta t\, \Delta\omega \geq 0.5$
- Performance is similar to GaussNet on Average
  - Best results is not reliable; How many runs?
    - Once I got 16.6% for SincNet while on ave PER is around 17.4%
  - Freq response of both Real and Complex is identical

Model	Valid.%	Avg. Test%	Best Test%
Gabor-CNN-CTC [18]	-	18.8	18.5
SincNet [2]	-	17.2	-
GaborReal	15.2	17.2	16.9
GaborComplex	15.2	17.1	16.7

CGCNN Advantages / Performance

- Optimal time-frequency resolution trade-off
  - Gaussian → $\Delta t\, \Delta\omega = 0.5$; For others → $\Delta t\, \Delta\omega \geq 0.5$
- Performance is similar to GaussNet on Average
  - Best results is not reliable; How many runs?
    - Once I got 16.6% for SincNet while on ave PER is around 17.4%
  - Freq response of both Real and Complex is identical

Model	Valid.%	Avg. Test%	Best Test%
Gabor-CNN-CTC [18]	-	18.8	18.5
SincNet [2]	-	17.2	-
GaborReal	15.2	17.2	16.9
GaborComplex	15.2	17.1	16.7

We called it GaussNet in our Interspeech 2019

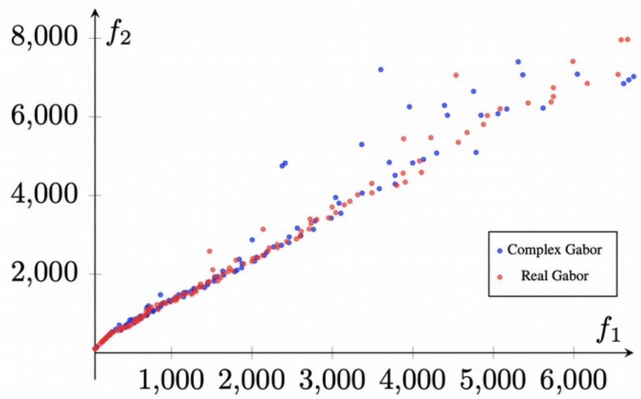
CGCNN other Advantages

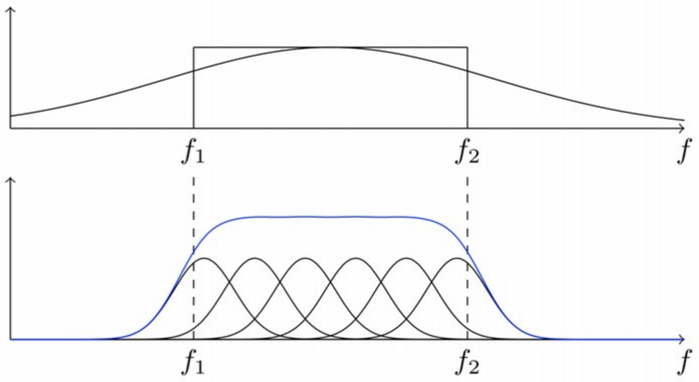
- *“But using complex quadratic filters that produce analytic signal for which the complex Gabor filtered signal is an approximation could help for instantaneous frequency estimation [23] and preserves the phase information that can be useful for other tasks such as speaker recognition.”*
  - Gabor filter is not *quadratic* → Should say *quadrature*!
  - Gabor filter *approximates analytic* signal
    - Gabor pair is not quadrature per sei because of DC component
  - Instantaneous frequency estimation → Relevance?
  - Preserve phase info … useful ... speaker recognition → Really?

Interpretation

- f_1 and f_2 are almost along a straight line → Constant Q
  - Biological plausibility

- GaussNet vs SincNet
  - GaussNet cannot model wide filters!
  - Second layer can compensate for this by combining narrow filters(?)





Complex CNN and MLP

- *We propose to fully take this complex representation into consideration by further processing it with complex-valued neural networks layers only.*
  - *Link to complexmodels in Github*

- I think by complex neural net they mean a quaternion kind of network with only two streams, instead of 4.

```
ge = torch.cos(2*math.pi*f_times_t*self.sample_rate)
ge = torch.mul(self.gaussian_window(self.n_, sigma), ge)

go = torch.sin(2*math.pi*f_times_t*self.sample_rate)
go = torch.mul(self.gaussian_window(self.n_, sigma), go)

max_, _ = torch.max(ge, dim=1, keepdim=True)
ge = ge / max_

max_, _ = torch.max(go, dim=1, keepdim=True)
go = go / max_

filters_ge = (ge * self.window_).view(self.out_channels, 1, self.kernel_size)
filters_go = (go * self.window_).view(self.out_channels, 1, self.kernel_size)

#filters_ge = ge.view(self.out_channels, 1, self.kernel_size)
#filters_go = go.view(self.out_channels, 1, self.kernel_size)

self.filters = torch.cat((filters_ge, filters_go),0)

conv_out = F.conv1d(waveforms, self.filters, stride=self.stride, padding=self.p

return conv_out
```

E2E-SINCNET: TOWARD FULLY END-TO-END SPEECH RECOGNITION

Titouan Parcollet[⋆‡†] *Mohamed Morchid*[⋆] *Georges Linarès*[⋆]

[⋆] Avignon Université, France
[‡] University of Oxford, UK
[†] Orkis, France

ICASSP2020
Barcelona

E2E-SincNet

- SincNet + Joint CTC-attention
  - SincNet + (B)RNN En-De + Attention + CTC
- Performance: WSJ → 4.7%
- Challenge:
  - Alignment between raw speech samples and characters in RNN En-De framework

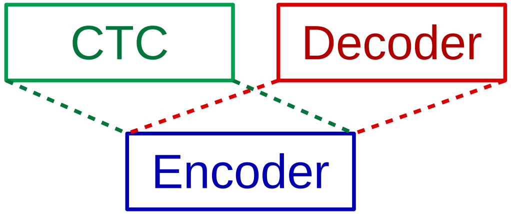
Joint CTC-Attention

- Advantages
  - Powerful seq2seq model
  - CTC → left-to-right alignment
  - Faster learning & convergence
- Shared encoder trained by L_{Joint}
- $\lambda_{Optimal}$? Depends ...
  - $\lambda_{Optimal} \approx 0.2$

$$L_{CTC} = -\sum_{(X,Y)\in D} \log P(Y|X)$$

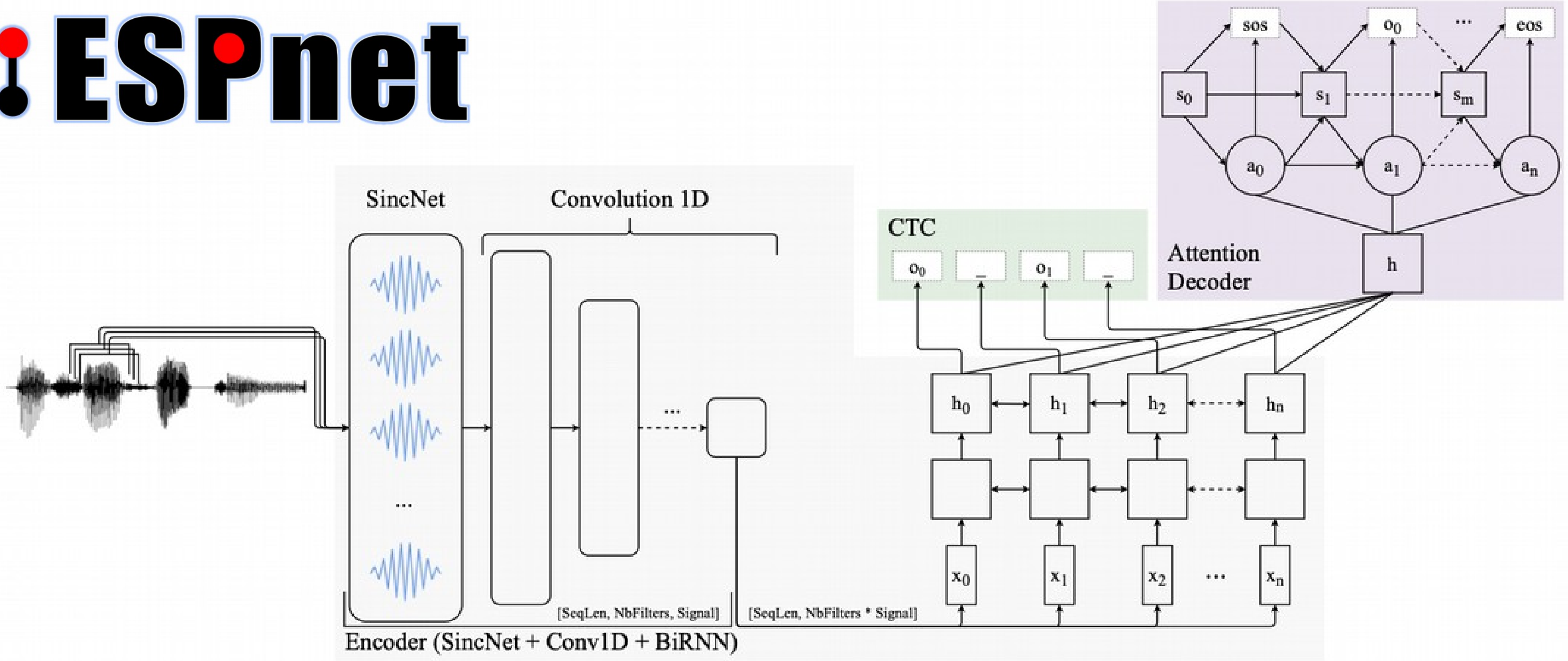
$$L_{En-De} = -\sum_{t} \log P(y_t|x_t, y_{t-1}^{truth})$$

$$L_{Joint} = \lambda \boxed{L_{CTC}} + (1-\lambda)\boxed{L_{En-De}}$$



E2E-SincNet Architecture

ESPnet



Github page

Experimental Setup

- Frame blocking for raw wavefor model
  - 25/10 ms instead of 200/10 ms
  - RNN models the context; No need to long frames!
- SincNet $\rightarrow$ $N_{filters}$=512, $L_{filters}$=129
  - Original setting in paper $\rightarrow$ $N_{filters}$=80, $L_{filters}$=251
  - Original setting in PyTorch Kaldi $\rightarrow$ $N_{filters}$=128, $L_{filters}$=129
- Other setting:
  - #epochs: 15 for TIMIT, 20 for WSJ; Optimiser: AdaDelta; No drop-out
  - λ_{TIMIT} = 0.5, λ_{WSJ}=0.2

Results on TIMIT and WSJ

- TIMIT (PER)
  - E2E CNN: 21.1%
  - E2E SincNet: **19.3**

Models	Fea.	Valid. %	Test %
E2E-CNN	RAW	18.9	21.1
ESPnet (VGG) [18]	FBANK	17.9	20.5
E2E-SincNet	**RAW**	**17.3**	**19.3**

- WSJ (PER)
  - E2E CNN: 6.5%
  - E2E SincNet: **4.7**%

Models	Fea.	Valid.	Test
BiGRU-Att. [9]	FBANK	-	9.3
Wav2Text [28]	FBANK	12.9	8.8
Jasper [8]	FBANK	9.3	6.9
E2E-CNN	RAW	9.8	6.5
ESPnet (VGG) [18]	FBANK	9.7	6.4
CNN-GLU-ASG [7]	RAW	8.3	6.1
SelfAttention-CTC [12]	FBANK	8.9	5.9
E2E-SincNet	**RAW**	**7.8**	**4.7**

Some Typos ...

- *"… It is also important to notice that g [filter impulse response] is* ***smoothed*** *based on the Hamming window …"*
  - *Multiplying in window → resolution-leakage trade-off*
  - *Convolving with window → smoothing* ← understandable ...
- *"In the original SincNet proposal [16], chunks of raw signal are cre- ated every* ***400ms*** *with a 10ms overlapping."* ↔ ***200ms***
- *In [16], the authors introduced a SincNet layer composed of* ***128*** *filters of size 251.* ↔ ***80*** (in PyTorch-Kaldi setup is 128)

LEARNING FILTERBANKS FROM RAW SPEECH FOR PHONE RECOGNITION

Neil Zeghidour[1,2], *Nicolas Usunier*[1], *Iasonas Kokkinos*[1], *Thomas Schatz*[2], *Gabriel Synnaeve*[1], *Emmanuel Dupoux*[2]

[1] Facebook A.I. Research, Paris, France, New York, USA
[2] CoML, ENS/CNRS/EHESS/INRIA/PSL Research University, Paris, France

End-to-End Speech Recognition From the Raw Waveform

Neil Zeghidour[1,2], *Nicolas Usunier*[1], *Gabriel Synnaeve*[1], *Ronan Collobert*[1], *Emmanuel Dupoux*[2]

[1] Facebook A.I. Research, Paris, France; New York & Menlo Park, USA
[2] CoML, ENS/CNRS/EHESS/INRIA/PSL Research University, Paris, France
{neilz, usunier, gab, locronan}@fb.com, emmanuel.dupoux@gmail.com

ATTENTION !!!

- These two work by *Zeghidour et al.* are actually *nonparametric* CNNs, initialised by parametric filters. That is,
  - First conv layer filters are initialised using Gammatone (GT) or Gabor which are parametric filters with two or three parameters
  - **BUT** number of free parameters during training equals filter length, i.e all filter taps are learnt ↔ non-parametric

- This is similar to Google work by *Hoshen et al.* and *Sainath et al.*
  - First conv layer init. by GT filters but learnt in a nonparametric fashion
  - Please refer to the 3rd tutorial in Listen! on 4/Feb/2020 for more details

LEARNING FILTERBANKS FROM RAW SPEECH FOR PHONE RECOGNITION

Neil Zeghidour[1,2], *Nicolas Usunier*[1], *Iasonas Kokkinos*[1], *Thomas Schatz*[2],
Gabriel Synnaeve[1], *Emmanuel Dupoux*[2]

[1] Facebook A.I. Research, Paris, France, New York, USA
[2] CoML, ENS/CNRS/EHESS/INRIA/PSL Research University, Paris, France

ICASSP 2018

Idea

- Replace MFSC* with **Time Domain (TD)** Filterbank
- Triangular filters are initially approximated by *Gabor wavelet*
- The filters are complex in time domain
  - Magnitude (modulus) is computed through L_2-pooling
    - DNN is not complex like CGCNN
- Learn the all filter taps, **NOT** f_c and *BW,* via backprop

* Mel Frequency Spectral Coefficient

Time-Domain Filterbank

- Consider MFSC (simply filterbank features ;-))

STFT of x at frame t

Nth filter freq response

$$MFSC_x(t,n) = \frac{1}{2\pi}\int_{\omega} |X(t,\omega)|^2 \; |\Psi_n(\omega)|^2 d\omega$$

Parseval's Theorem

$$MFSC_x(t,n) = \sum_{\tau} \left(x_t(\tau) * \psi_n(\tau)\right)^2$$

Time-Domain Filterbank

- Approximate MFSC ...

$$MFSC_x(t,n) = \frac{1}{2\pi}\int_{\omega} |X(t,\omega)|^2 \; |\Psi_n(\omega)|^2 d\omega$$

$$MFSC_x(t,n) = \sum_{\tau} \left(x_t(\tau) * \psi_n(\tau)\right)^2$$

Approximate

$$|\Phi_n(\omega)|^2 \approx |\Psi_n(\omega)|^2$$

$$MFSC_x(t,n) \approx |x * \varphi_n|^2 * |\phi|^2(t)$$

Time-Domain Filterbank

- Approximate MFSC ...

$$MFSC_x(t,n) = \frac{1}{2\pi}\int_\omega |X(t,\omega)|^2 \, |\Psi_n(\omega)|^2 d\omega$$

$$MFSC_x(t,n) = \sum_\tau \left(x_t(\tau) * \psi_n(\tau)\right)^2$$

Approximate

$$MFSC_x(t,n) \approx |x * \varphi_n|^2 * |\phi|^2(t)$$

$$|\Phi_n(\omega)|^2 \approx |\Psi_n(\omega)|^2$$

Time-Domain Filterbank

- Approx. MFSC with (first-order) Scattering Spectrum

$$MFSC_x(t,n) = \sum_{\tau} (x_t(\tau) * \psi_n(\tau))^2$$

$$Mx(t,n) \approx |x * \varphi_n|^2 * |\phi|^2(t)$$

φ$_n$(t) wavelet approximates *n*th (triangular) filter

Hanning window (scaling function)

For averaging or smoothing

Φ should not be shorter than *φ$_n$(t)*

$$\varphi_n(t) \propto \frac{1}{\sqrt{2\pi}\sigma_n} \exp(-\frac{t^2}{2\sigma_n^2}) \exp(-2\pi i \eta_n t)$$

Time-Domain Filterbank

- Approx. MFSC with (first-order) Scattering Spectrum

$$MFSC_x(t,n) = \sum_{\tau} (x_t(\tau) * \psi_n(\tau))^2$$

$$Mx(t,n) \approx |x * \varphi_n|^2 * |\phi|^2(t)$$

$\omega_n \rightarrow$ FWHM:
full width at half maximum
Simply -3dB bandwidth ;-)

$$\varphi_n(t) \propto \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{t^2}{2\sigma_n^2}\right) \exp(-2\pi i \eta_n t)$$

$$\sigma_n = \frac{2\sqrt{2log2}}{\omega_n}$$

η_n: f_c of n^{th} filter

Time-Domain Filterbank

- Approx. MFSC with (first-order) Scattering Spectrum

$$MFSC_x(t,n) = \sum_{\tau} (x_t(\tau) * \psi_n(\tau))^2$$

$$Mx(t,n) \approx |x * \varphi_n|^2 * |\phi|^2(t)$$

$$\varphi_n(t) \propto \frac{1}{\sqrt{2\pi}\sigma_n} \exp(-\frac{t^2}{2\sigma_n^2}) \; \exp(-2\pi i \eta_n t)$$

φ_n is normalised to have the same energy as Ψ_n

Time-Domain Filterbank

- Approx. MFSC with (first-order) Scattering Spectrum

$$MFSC_x(t,n) = \sum_{\tau} (x_t(\tau) * \psi_n(\tau))^2$$

$$Mx(t,n) \approx |x * \varphi_n|^2 * |\phi|^2(t)$$

$$\varphi_n(t) \propto \frac{1}{\sqrt{2\pi}\sigma_n} \exp(-\frac{t^2}{2\sigma_n^2}) \ \exp(-2\pi i \eta_n t)$$

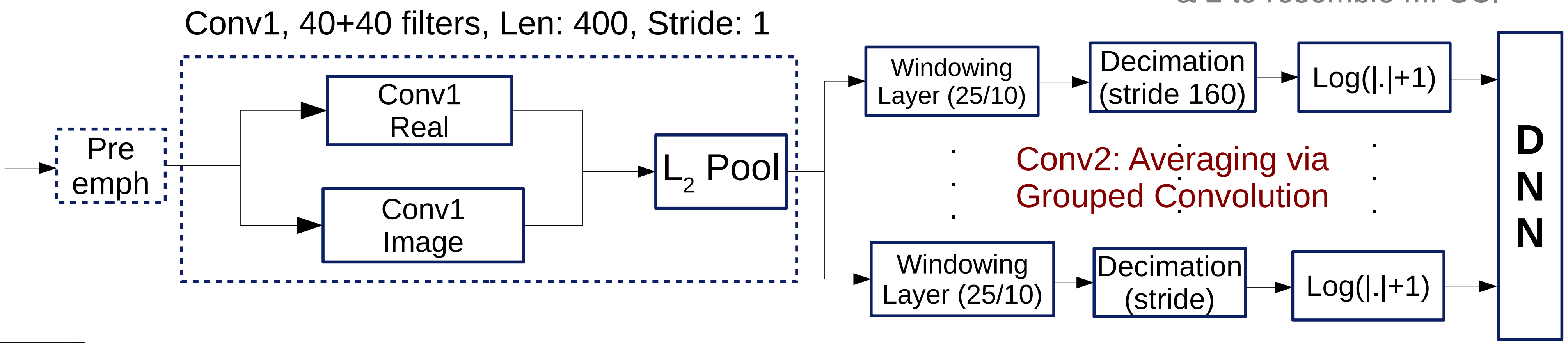
Second-order

$$||x * \varphi_{n1}| * \varphi_{n2}| * |\phi|^2(t)$$

TD-Filterbank System Architecture

- DNN1: 5 layers CNN (ReLU), 1k filters, width 5, Do 0.5
- DNN2: DNN1 + dropout (Do) 0.7
- DNN3: 8 layers,CNN, PReLU, Do 0.7

Biases set to zero for Conv1 & 2 to resemble MFSC.



#groups = #channels

TD-Filterbank Types

- Fixed
  - Init with Mel-fbank fc/BW; freeze fbank (φ) and ave (Φ) during training
- Learn-filterbank
  - Init with Mel-fbank fc/BW; learn fbank, freeze ave in hann^2
- Randinit
  - Init randomly; learn both fbank/ave
- Learn-all
  - Init with Mel-fbank fc/BW; learn both fbank (φ) and averaging (Φ)

Experimental Results – E2E

- Comparable PER to MFSC
  - Marginal gain
- Hanning^2 ave is good enough
  - No need to learn averaging!
- Initialisation is important
  - Randinit performs poorly!
  - Data size, TIMIT?

TIMIT

Learning mode	Dev PER	Test PER
MFSC	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

Architectures:
DNN2: CNN-5L-ReLU-do0.7

Experimental Results
E2E phone Recognition

	Model	Input		Dev PER	Test PER
	Hybrid HMM/Hierarchical CNN + Maxout + Dropout [10]	MFSC + energy + Δ + $\Delta\Delta$		13.3	16.5
	CNN + CRF on raw speech [15]	wav		-	29.2
	Wavenet [16]	wav		-	18.8
	CNN-Conv2D-10L-Maxout [17]	MFSC		16.7	18.2
	Attention model + Conv. Features + Smooth Focus [18]	MFSC + energy + Δ + $\Delta\Delta$		15.8	17.6
	LSTM + Segmental CRF [19]	MFSC + Δ + $\Delta\Delta$		-	18.9
	LSTM + Segmental CRF [19]	MFCC + LDA + MLLT + MLLR		-	17.3
DNN1	CNN-5L-ReLU-do0.5	MFSC		18.4	20.8
	CNN-5L-ReLU-do0.5 + TD-filterbanks	wav		18.2	20.4
DNN2	CNN-5L-ReLU-do0.7	MFSC	40 filters	17.8	20.6
	CNN-5L-ReLU-do0.7 + TD-filterbanks	wav	40+40	17.3	20.3
DNN3	CNN-8L-PReLU-do0.7	MFSC		16.2	18.1
	CNN-8L-PReLU-do0.7 + TD-filterbanks	wav		15.6	18.1
	CNN-8L-PReLU-do0.7 + TD-filterbanks-Learn-all-pre-emp	wav		15.6	18.0

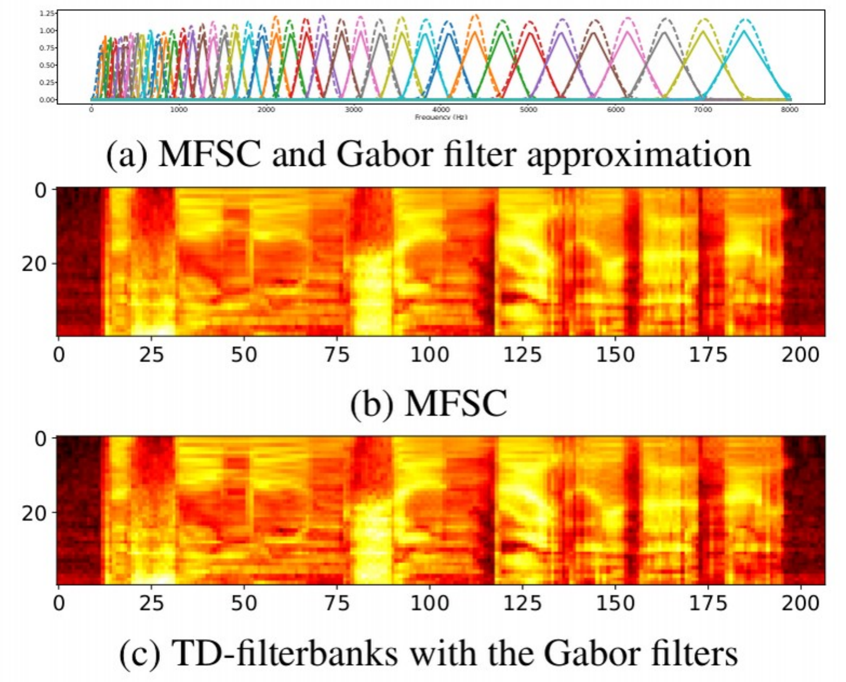
PReLU: Parametric ReLU; learn slope for negative pre-activ

Learn pre-emphasis (FIR high-pass, $1\text{-}az^{-1}$)

- DNN3 is better that both DNN1 == DNN2
- Comparable performance to MFSC
- Learn pre-emphasis → 0.1% PER reduction

Gabor Filters vs Triangular Filters

- Mel fbank (solid) vs Gabor (dashed)
  - Gabor is smoother
    - Gaussian vs Triangle
- Spectrograms are similar

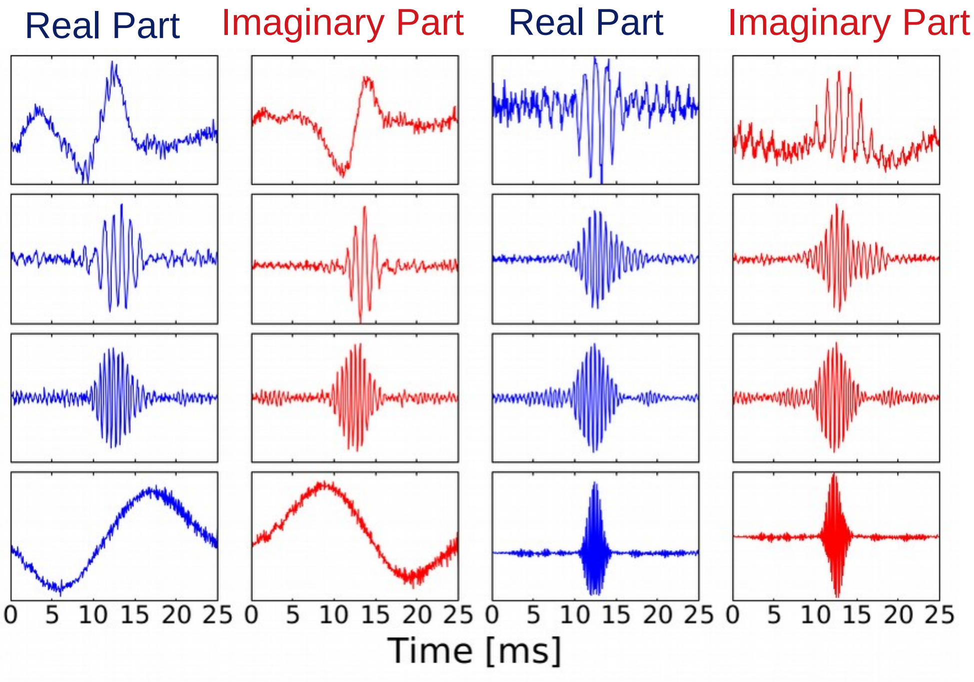


(a) MFSC and Gabor filter approximation
(b) MFSC
(c) TD-filterbanks with the Gabor filters

Learned Filters

- Filters are biologically plausible ...
  - ✔ Asymmetric
  - ✔ Sharp attack and slow decay

- Spread of filters in time & freq domains could be different

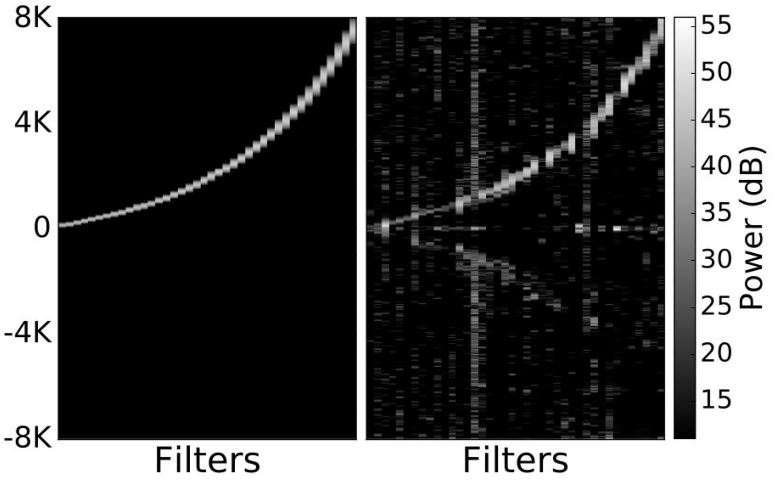


CNN-8L- PReLU-do 0.7 + TD-filterbanks model

Learned Filters

- f_c remain similar to Mel; BW varies a lot
- Energy@negative freq?
  - ✓ Yes, complex filter and Re/Im parts are not Hilbert pair
  - ✓ Initially, Re & Im were ~ Hilbert pair (Gabor)
  - ✓ Analyticity is not preserved during training
- Importance of preserving analyticity?
  - ➢ sub-band Hilbert envelop extraction
- r_a = E@Neg/E@Pos → r_a=0.26
  - ➢ Analytic signal → r_a=0; Real Signal → r_a=1

CNN-8L- PReLU-do 0.7 + TD-filterbanks model



End-to-End Speech Recognition From the Raw Waveform

Neil Zeghidour[1,2], *Nicolas Usunier*[1], *Gabriel Synnaeve*[1], *Ronan Collobert*[1], *Emmanuel Dupoux*[2]

[1] Facebook A.I. Research, Paris, France; New York & Menlo Park, USA
[2] CoML, ENS/CNRS/EHESS/INRIA/PSL Research University, Paris, France
{neilz, usunier, gab, locronan}@fb.com, emmanuel.dupoux@gmail.com

INTERSPEECH 2018

E. Loweimi

Study the Effect of Following ...

- Gammatone instead of Gabor (~ Scattering Spec)
- Importance of *low-pass filtering*
  - Hanning^2 window vs max-pooling
- Importance of *instance normalisation*
  - Mean-var norm per channel per utterance [after log]

SCattering vs GammaTones Models

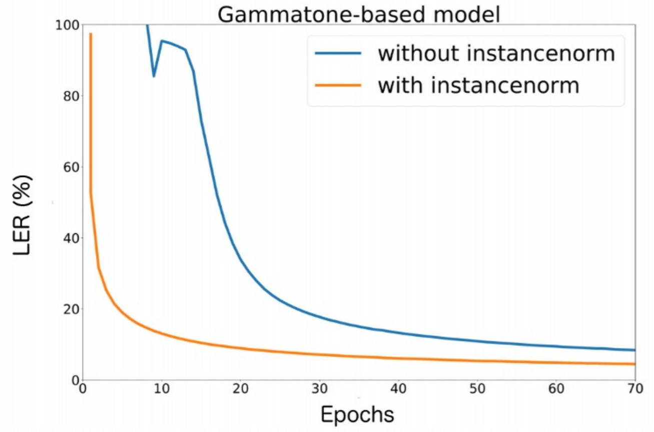
- Both are parametric CNNs
- Differences
  - Belong to different families
  - SC is complex; GT is real
  - #filters → SC: 40+40; GT: 40
  - Non-linearity → $|L_2|^2$ vs ReLU
  - Pooling → $|\text{Hann}|^2$ vs Max-pooling

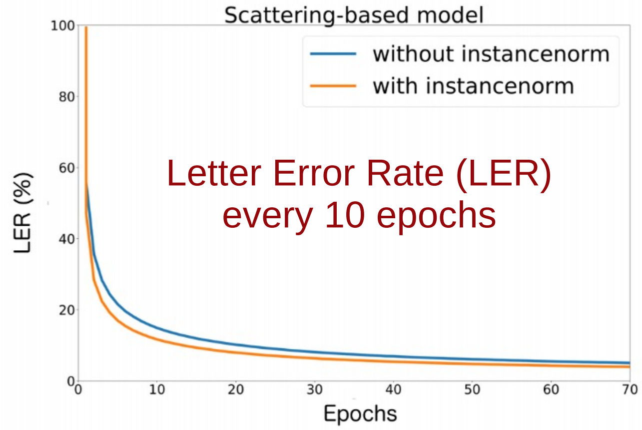
	SCATTERING	GAMMATONES
Conv[1] (*#in-#out-width-stride*)	1-80-400-1	1-40-400-1
non-linearity	sq. L2-Pooling	ReLU
low-pass filter (*wdth*=400, *strd*=160)	sq. Hanning	max-pooling or sq. Hanning
log-compression[2]	log(1+abs(.))	log(0.01+abs(.))
normalization	mean-var. per-channel per-sentence	

400 samples width ≡ 25 ms

Instance Normalisation

- MVN per channel per utterance
- For Gammatone-based models
  - Critical, Faster convergence, Stabilises training
- For Scattering-based models
  - Minor effect, Y? Scaling func.
  - Slightly faster convergence





Experimental Setup

- Framework: End-to-End, WSJ
- Toolkit: Wav2letter → Facebook toolkit for E2E ASR
- Training: SI284, Dev: Nov93-dev, Test: Nov92-eval
- Performance measure: WER and LER
- Architecture: 16 layers CNN with GLU (Gated Linear Unit)
  - GLU: halves #output-channels (half act as gate)
- LM for WER → standard 4-gram built on WSJ LM data

Initialisation & Low-pass Filter Effect

- Gamma & Scatt outperform mel-fbank
- Initialisation effect (Nov92-Eval)
  - GT → GT init better than rand
  - SC → rand init better than Gabor/Mel
- Low-pass effect (Nov92-Eval)
  - GT → *Han-fixed* better than *max-pool*
  - SC → *Han-fixed* better than *Han-learnt*

FRONT END	FILTER INIT	LOW-PASS	NOV93-DEV LER	NOV93-DEV WER	NOV92-EVAL LER	NOV92-EVAL WER
mel-fbanks			6.9	9.5	4.9	6.6
gamm (learnt)	gamm	Han-fixed	6.9	9.1	4.9	5.9
		max-pool	7.2	9.3	4.9	6.0
	rand	Han-fixed	7	8.9	4.9	5.9
		max-pool	7.2	9.2	5.1	6.3
scatt (learnt)	scatt	Han-fixed	6.7	8.3	4.6	6.1
		Han-learnt	6.7	8.9	4.5	6.3
	rand	Han-fixed	6.8	8.5	4.7	5.7
		Han-learnt	6.9	8.9	4.9	5.8

Dev Eval

Effect of Learning Pre-emphasis

- Pre-emphasis filter
  - FIR, 2 taps [-0.97, 1], highpass
  - Conv layer, kWidth=2, Stride=1
  - Learn it; init with [-0.97, 1]
- Helpful for both GT and SC
  - GT → 0.1 – 0.2
  - SC → -0.4 – 0.4

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

Dev Eval

Effect of Learning Pre-emphasis

- Pre-emphasis filter
  - FIR, 2 taps [-0.97, 1], highpass
  - Conv layer, kWidth=2, Stride=1
  - Learn it; init with [-0.97, 1]
- Helpful for both GT and SC
  - GT → 0.1 – 0.2
  - SC → -0.4 – 0.4
- WER & LER correlation can be < 0

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

Corr < 0 Corr > 0

ACOUSTIC MODEL ADAPTATION FROM RAW WAVEFORMS WITH SINCNET

Joachim Fainberg, Ondřej Klejch, Erfan Loweimi, Peter Bell, Steve Renals

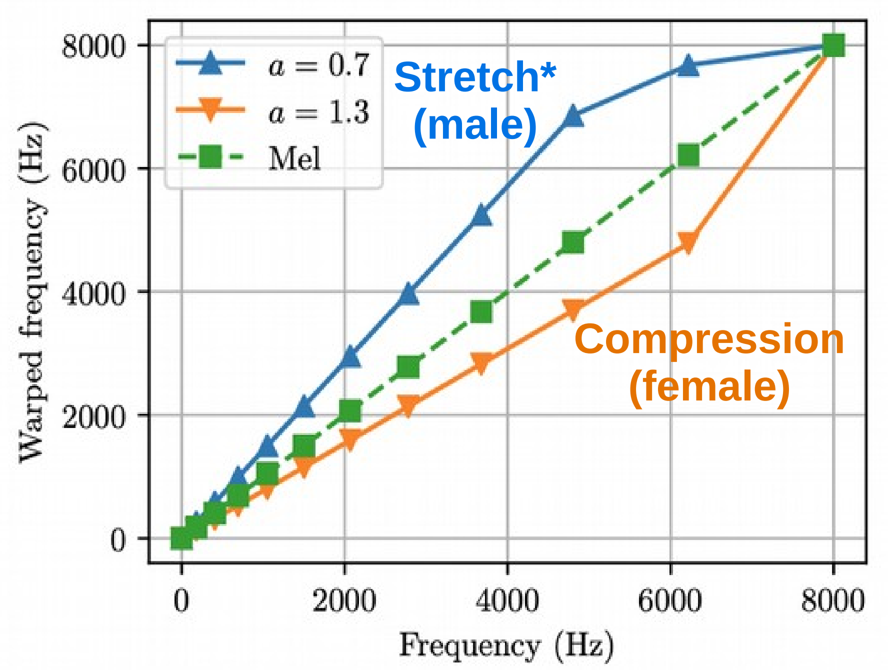
Centre for Speech Technology Research, University of Edinburgh, United Kingdom

IDEA ...

- Raw waveform acoustic model adaptation
  - Adapt parameters of Sinc layer, i.e. f_c and BW
  - Compared with VTLN and LHUC
- How:
  - Trained on adult (AMI-ihm), 100 hours, meeting speech
  - Adapted to children (PF-STAR), 14 hours, read speech

SincNet Adaptation vs VTLN

- Parameters
  - VTLN → $f_{warping}(\omega, \alpha)$ ← 1 param
  - SincNet → f_c & BW ← 2#filters
- Domain
  - SincNet ↔ time
  - VTLN ↔ frequency
- Learn f_c & BW vs grid search for α



Note*: In HTK α > 1 ≡ stretch.

Warping function:
- Exp: piece-wise linear, bilinear, etc.
- Characterised by α

Experimental Setup

- Architecture:
  - Sinc (40 filters) + 6L 1D CNN
  - 9M parameters
- Optimiser:
  - Adam, lr=0.0015
- Frame → 200ms / 10ms
- Implementation:
  - Keras + TF
- LM interpolation of AMI (KN-3gram+Fisher) and PF-STAR

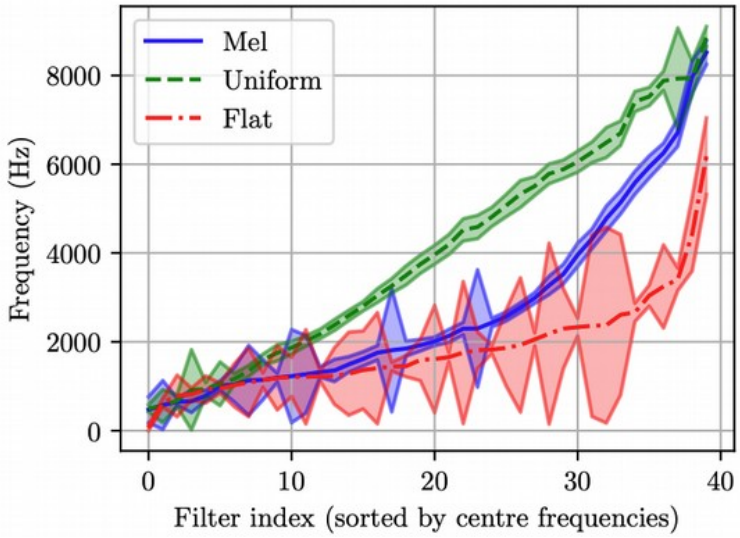
#	Type	Dim	Size	Dil	Params
1	SincConv	40	129	-	80
-	MaxPooling	-	3	-	
2	BN(ReLU(Conv))	800	2	1	68,000
-	MaxPooling	-	3	-	
3	BN(ReLU(Conv))	800	2	3	1,284,000
-	MaxPooling	-	3	-	
4	BN(ReLU(Conv))	800	2	6	1,284,000
-	MaxPooling	-	3	-	
5	BN(ReLU(Conv))	800	2	9	1,284,000
-	MaxPooling	-	2	-	
6	BN(ReLU(Conv))	800	2	6	1,284,000
7	ReLU(Conv)	800	1	1	640,800
8	Softmax(Conv)	3976	1	1	3,184,776

Experimental Results on AMI

6 epochs

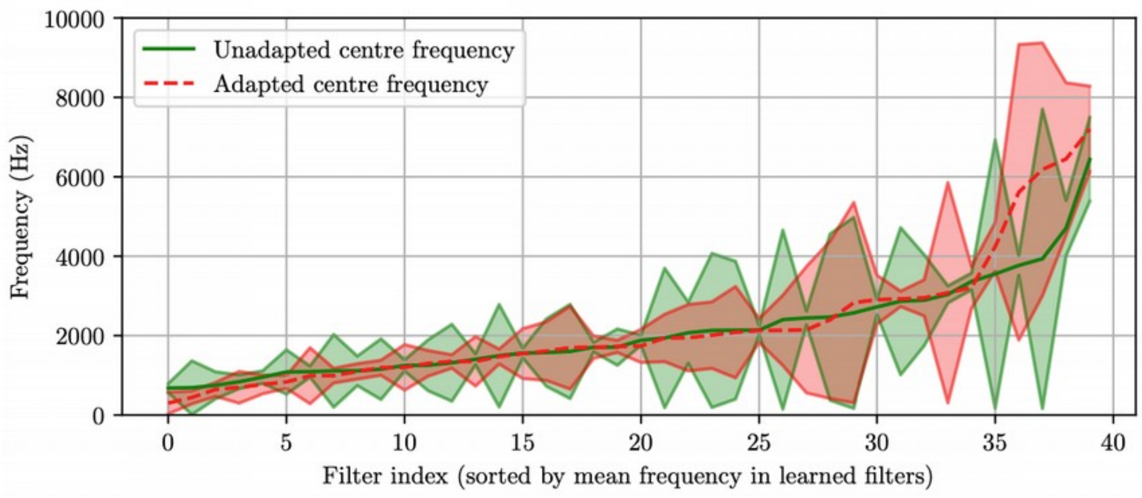
- Filter initialisation
  - Mel
  - Flat (uniform, not random)
  - Uniform (random)

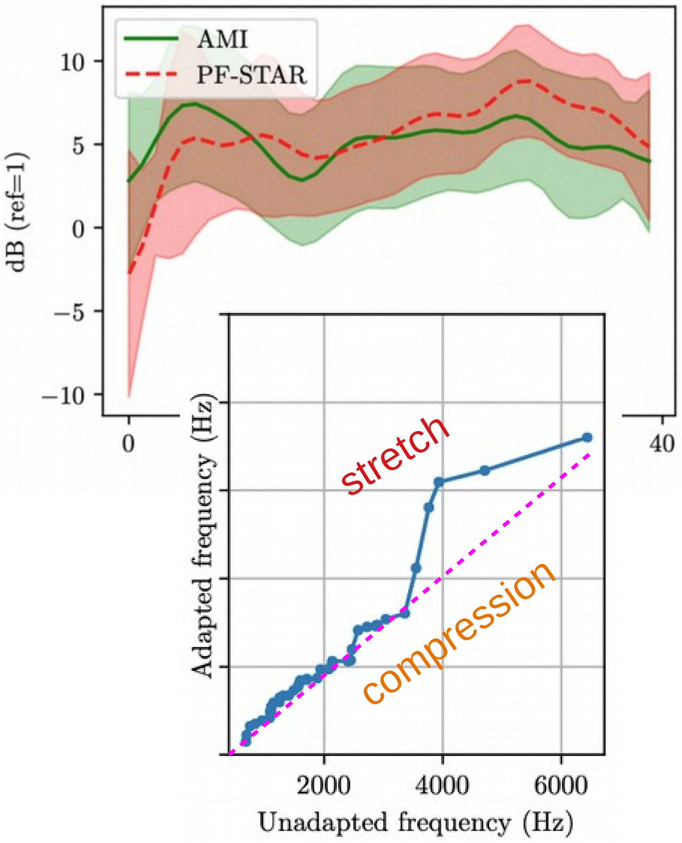
- Similar WER **BUT** markedly different learned f_c & *BW*



Initialisation	Eval	Dev
Mel	30.6	28.0
Flat	30.2	28.0
Uni	30.3	27.9

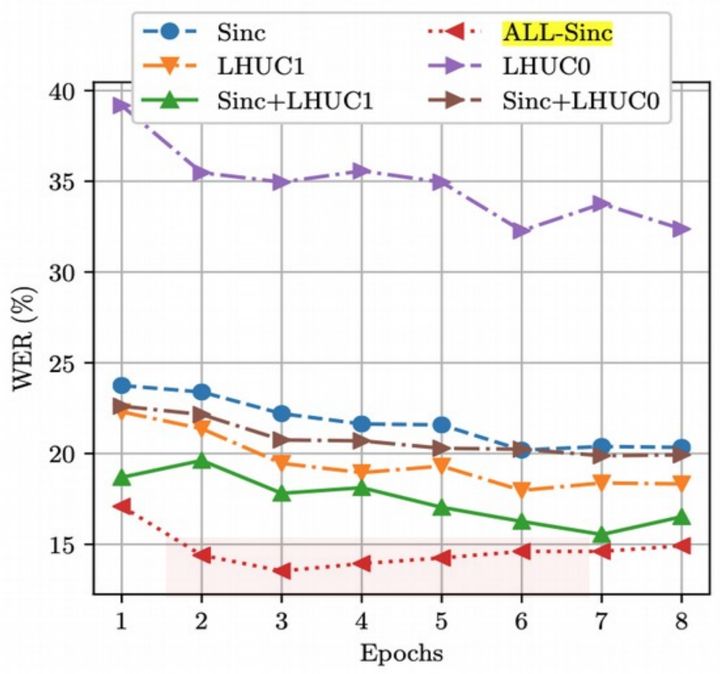
Adapted Sinc Layer (filterbank)





- Adaptation from Adult to Children ...
  - involves spectrum stretch, i.e $f_{adapted} > f_{unadapted}$
  - more energy in high frequencies

ASR Results



– LHUC adapts the filter gain
– All-Sinc prune to overfitting

Method	WER (%)	Params
Unadapted	59.06	-
Sinc	20.34	80
LHUC0	32.37	40
Sinc+LHUC0	19.93	120
LHUC1	18.33	800
Sinc+LHUC1	16.52	880
ALL-Sinc	14.92	$\sim$ 9M

– LHUC0: LHUC on Sinc layer
– Sinc+LHUC0: Adapt Sinc + LHUC0
– ALL-Sinc: all param, excluding sinc

Wrap-up

- Parametric CNN
  - Allows for embedding prior info in the network
  - Can improve the performance, even for small tasks
  - Faster convergence with fewer data
- Future work
  - Further E2E, Raw waveform + RNNs
  - Raw waveform + Unsupervised
  - Dynamic/Evolution of the first layer during training
  - ...

That’s it!

- Thanks for Your attention
- Q/A

- Appendices

  A1) Gammatone Filterbank

  A2) Denis Gabor Contributions

  A3) CTC

  A4) VTLN

A1) Gammatone Filterbank

- **Structure**: A set of IIR bandpass filters, defined in time domain
- **Obtained** by *reverse correlation* from measurements of auditory nerve responses of cats
- **Parameter** k: gain, B: decay factor, f_c: centre freq (Hz), n: order
- 3 < **n** < 5 → Good approximation for human auditory (Cochlea) filters
- $\mathbf{f_c}$ based on Greenwood or equal distance in a perceptual scale
- **B** → Equivalent Rectangular Bandwidth (Hz)

$$h_i(t) = k\ t^{n-1} \exp(-2\pi B_i t)\ \cos(2\pi f_i t + \phi)$$

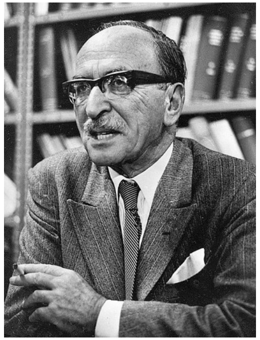
$$f_i^{gw} = 165.4(10^{2.1x} - 1)$$

$$0 < x < 1$$

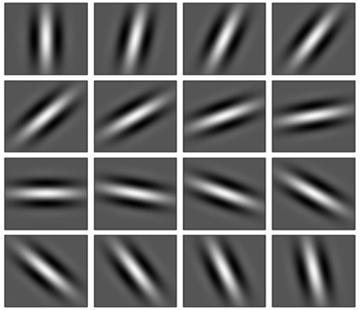
$$B_i = 1.019 * 24.7(4.37\ f_i/1000 + 1)$$

A2) Dennis Gabor’s Contributions

- Electrical Engineer and Physicist
  - Hologram → 1971 Nobel prize
- Signal Processing
  - Gabor-Heisenberg uncertainty principle ($\Delta t \, \Delta\omega \geq 0.5$)
  - Gabor filters
    - Texture analysis + perceptually motivated
  - Gabor Transform/Wavelet
    - FT + Gaussian window (1946) → STFT
    - Gabor atoms $g_{t_0,\omega_0} = g(t - t_0)\exp(j\omega_0 t)$



Dennis Gabor
(1900-1979)



A2) Gabor Transform Limits

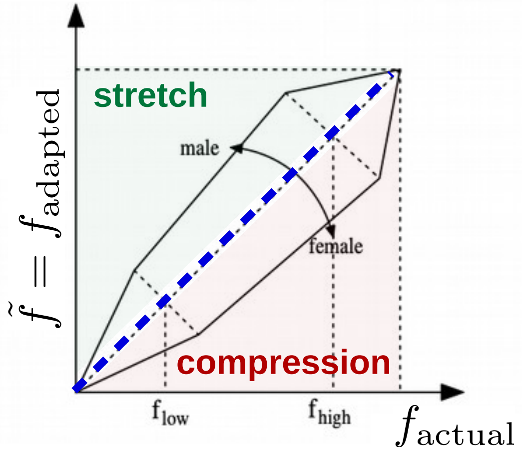
- Non-orthogonal family, though forms a *frame*
  - Complete but redundant
- Well-localised but infinite support
  - Truncation
- Gabor pair is not precisely quadrature
  - Because of DC component of even part
  - BUT approximately, it is

A3) **C**onnectionist **T**emporal **C**lassification

- CTC is a special output layer for Seq2Seq modes (RNNs)
- Handles Y_{len} != X_{len}; Y_{len} should be shorter
- Does not require lexicon and X→Y alignment
- Blank symbol → to handle all possible alignments
- Learned-based on likelihood (CE)
- Loss efficiently computable using forward/backward algorithms
- Decoding → beam search + Dynamic Programming
- Disadvantages
  - Conditional-independence assumption, i.e $y_t \perp\!\!\!\perp y_{t-1} \mid X$
  - Does not *explicitly* model inter-label dependencies

A4) Vocal Tract Length Norm. (VTLN)

- VT Length (VTL) variation shifts formants, almost linearly → $F_i \approx \dfrac{(2i-1)\text{v}}{4\,VTL}$
  - Female speakers, shorter VTL => larger formants
  - Adaptation for female to standard spk → compress spectrum, i.e. $\tilde{f} < f$
- VTLN HOW:
  - Choose warping function, e.g. piece-wise linear, bilinear
  - Find the warping factor
    1. First pass recognition
    2. Forced-alignment for all warping factors (grid search)
    3. Select factor with max likelihood
    4. Second pass recognition after applying optimal warping factor



- Effective when speakers clearly identifiable, e.g telephone speech