Deep Learning for *Audio Processing*

Geoffroy Peeters

contact: geoffroy.peeters@telecom-paris.fr
Télécom-Paris, IP-Paris, France

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

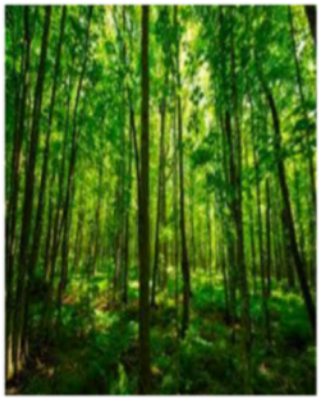
Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

Various categories of audio content
Specificities

	Speech (object)	Music (artwork)	Scene
Image			
Audio			
	Mostly **single** speaker A **structured** sound source	Multi timbre (texture) **polyphonic** **Structure** of sound sources **Dynamic** control (mixing)	Multi source **polyphonic** **Unstructured** sound sources

source : S. Dieleman, J. Pons, J. Lee, Waveform-based music processing with deep learning, tutoriel, ISMIR, 2019

Various categories of audio content
Applications

	Speech
Description	Speech to text (ASR) Speaker recognition Speaker diarization
Transformation	Speech separation, enhancement Speech coding Speech transformation
Generation	Text to speech

Speech-to-Text = Automatic Speech Recognition (ASR)

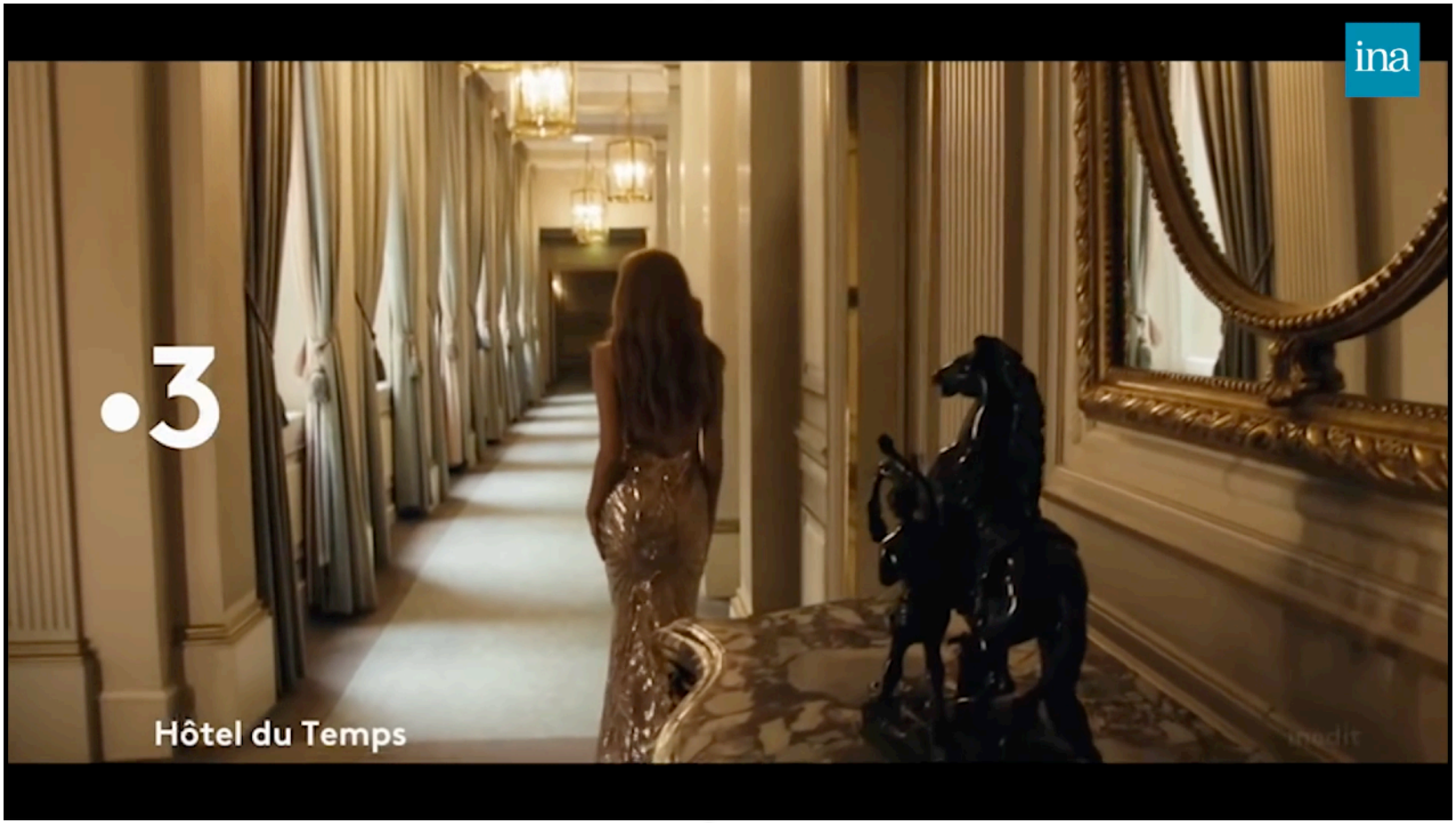
Microsoft :

- https://www.youtube.com/watch?time_continue=544&v=Nu-nlQqFCKg

Skype :

- https://www.youtube.com/watch?v=JrITzS7Fk6o

Speaker modification (Deep Fake)

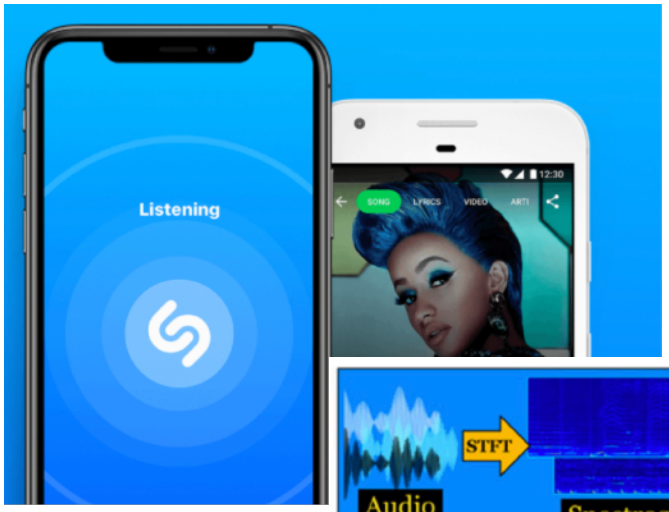


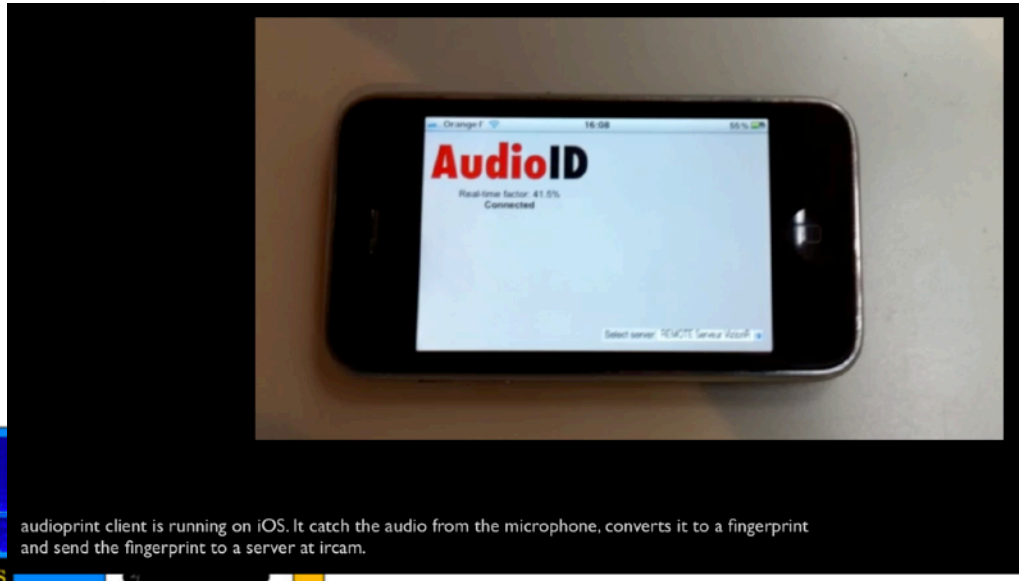
https://www.youtube.com/watch?v=seXKtSGf4U8

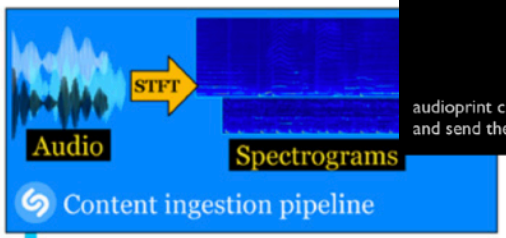
Various categories of audio content
Applications

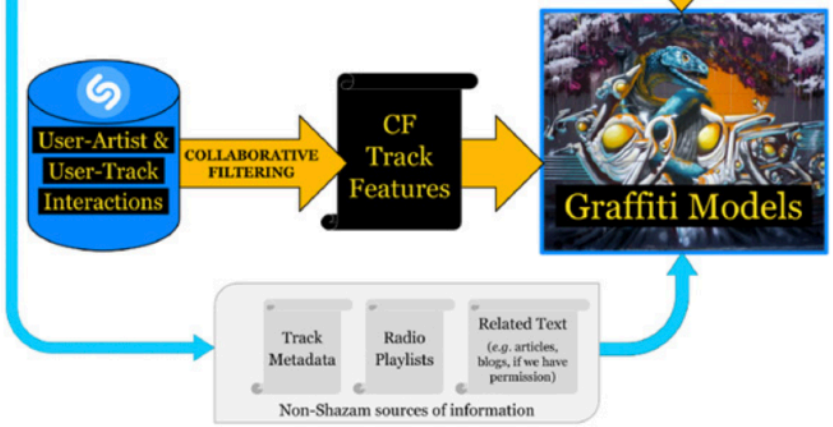
	Music
Description	**Audio ID (Shazam)** **Auto-tagging** **Recommendation** **Content-description (pitch, chord, tempo)** **Lyrics alignment/ recognition**
Transformation	**Singing separation (Karaoke)** **Unmixing** **Style transfer**
Generation	**Sound generation** **Music generation (OpenAI)**
	MIR http://ismir.net

Audio ID (Shazam like)

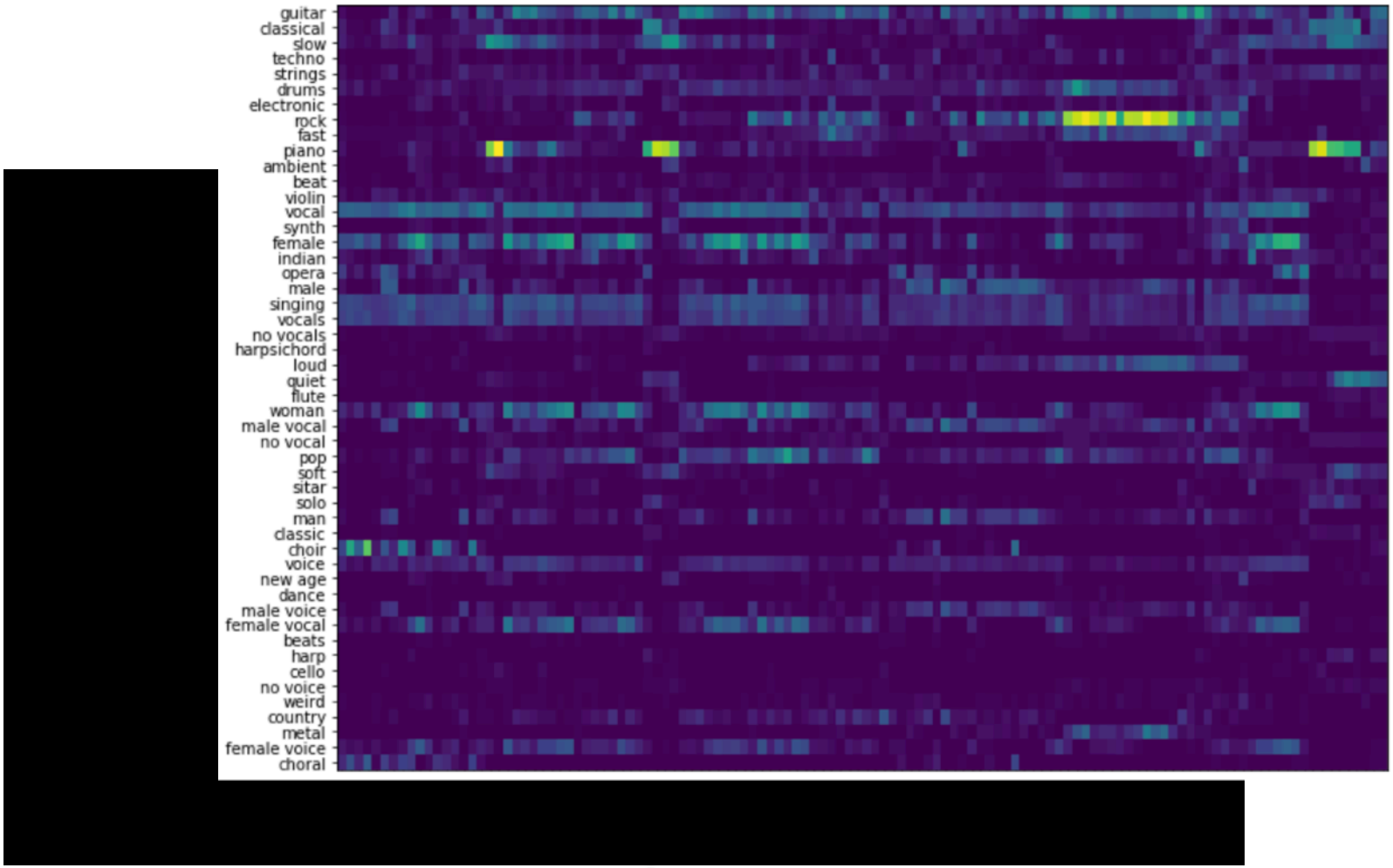








Music Auto-Tagging



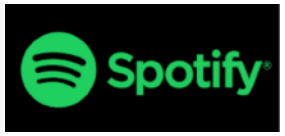
[J. Pons and X. Serra. "**Musicnn: Pre-trained convolutional neural networks ...**", In Late-Breaking/Demo Session of ISMIR, 2019]

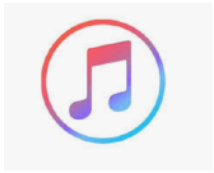
Music Applications

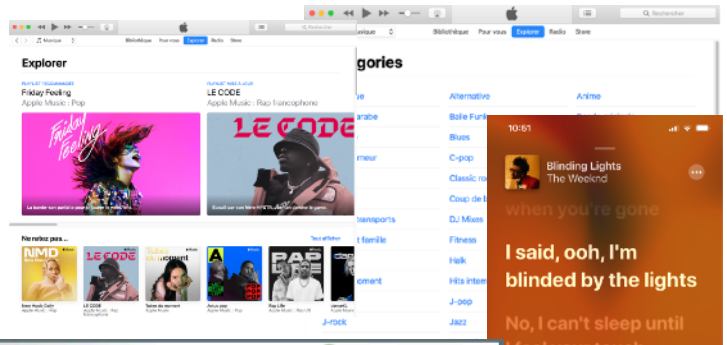
Music search engine

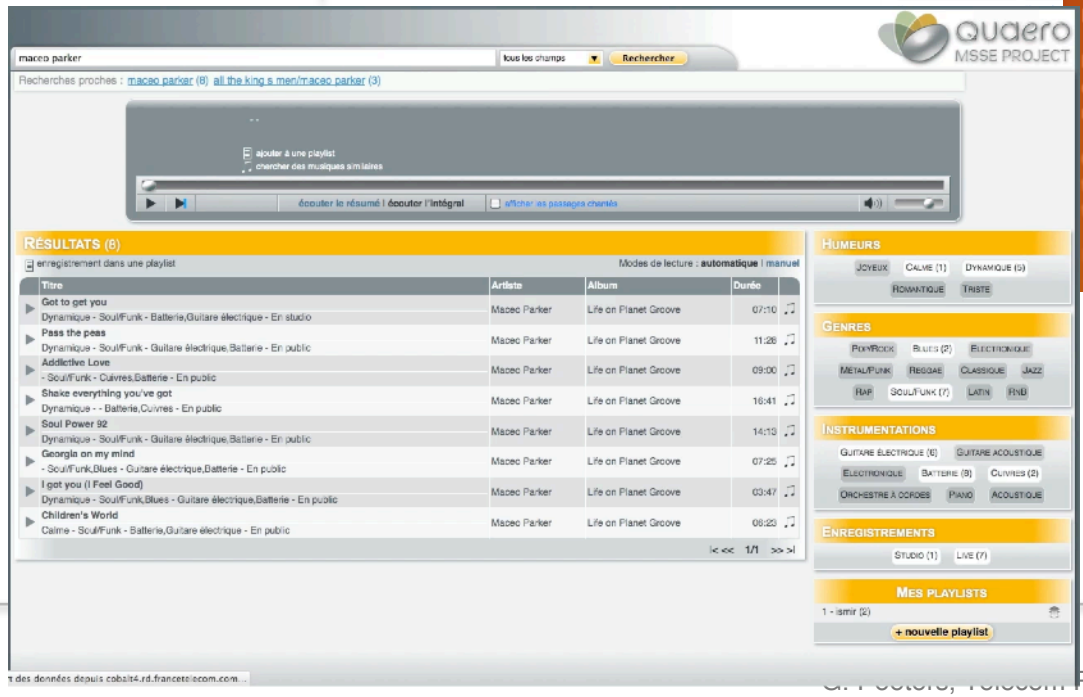
- **Streaming services:**
  - YouTube-Music, Pandora, Spotify, Apple Music, Deezer, ...

- Auto-tagging
- Lyrics synchronization
- Recommendation
  - Search-by-similarity
  - Collaborative Filtering





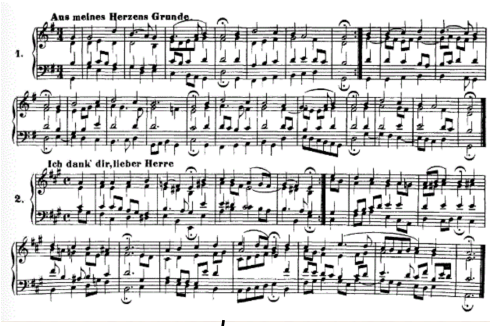




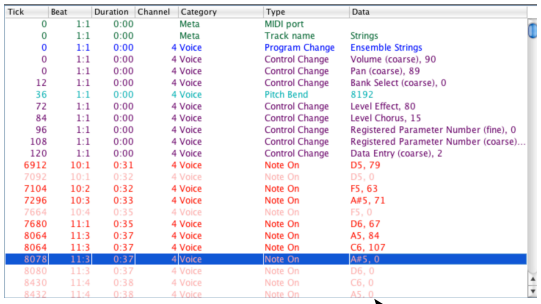
Music has various representations

Musique

Partition



MIDI

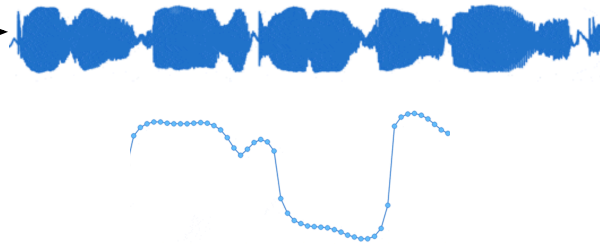


Tick	Beat	Duration	Channel	Category	Type	Data
0	1:1	0:00		Meta	MIDI port	
0	1:1	0:00		Meta	Track name	Strings
0	1:1	0:00	4 Voice	Program Change	Ensemble Strings	
0	1:1	0:00	4 Voice	Control Change	Volume (coarse), 90	
0	1:1	0:00	4 Voice	Control Change	Pan (coarse), 89	
12	1:1	0:00	4 Voice	Control Change	Bank Select (coarse), 0	
36	1:1	0:00	4 Voice	Pitch Bend	8192	
72	1:1	0:00	4 Voice	Control Change	Level Effect, 80	
84	1:1	0:00	4 Voice	Control Change	Level Chorus, 15	
96	1:1	0:00	4 Voice	Control Change	Registered Parameter Number (fine), 0	
108	1:1	0:00	4 Voice	Control Change	Registered Parameter Number (coarse)...	
120	1:1	0:00	4 Voice	Control Change	Data Entry (coarse), 2	
6912	10:1	0:31	4 Voice	Note On	D5, 79	
7092	10:1	0:32	4 Voice	Note On	D5, 0	
7104	10:2	0:32	4 Voice	Note On	F5, 63	
7296	10:3	0:33	4 Voice	Note On	A#5, 71	
7680	11:1	0:35	4 Voice	Note On	D6, 67	
8064	11:3	0:37	4 Voice	Note On	A5, 84	
8064	11:3	0:37	4 Voice	Note On	C6, 107	
8078	11:3	0:37	4 Voice	Note On	A#5, 0	
8080	11:3	0:37	4 Voice	Note On	D6, 0	
8430	11:4	0:38	4 Voice	Note On	C6, 0	
8432	11:4	0:38	4 Voice	Note On	A5, 0	

Transcription

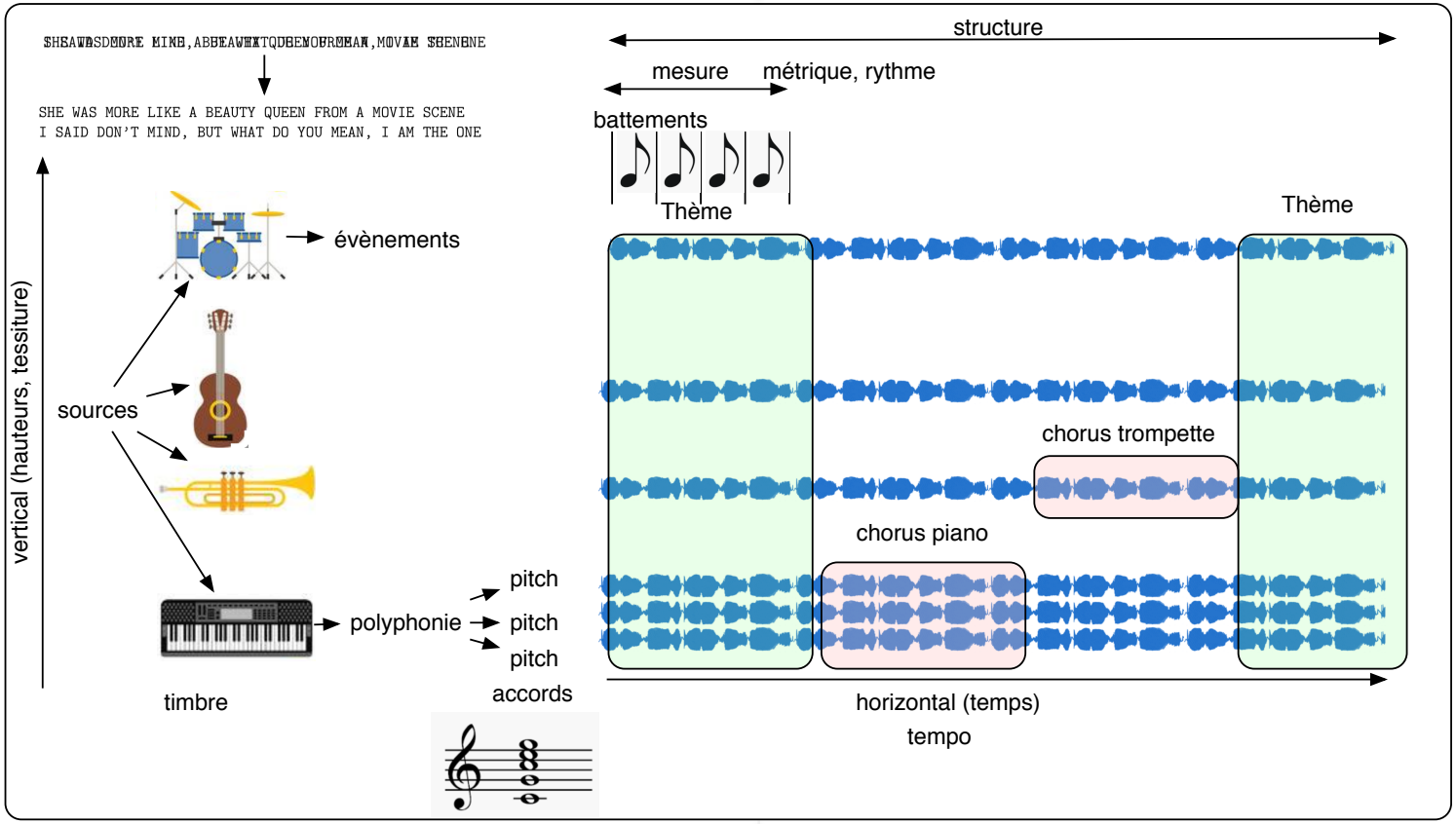
Enregistrement

Audio

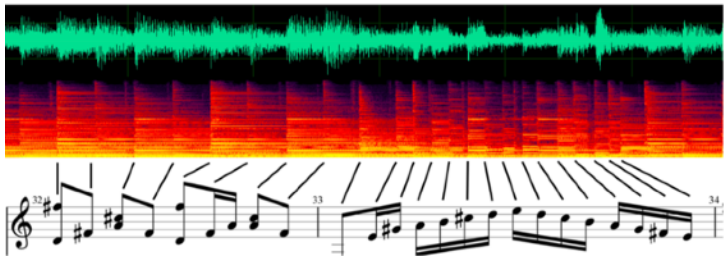


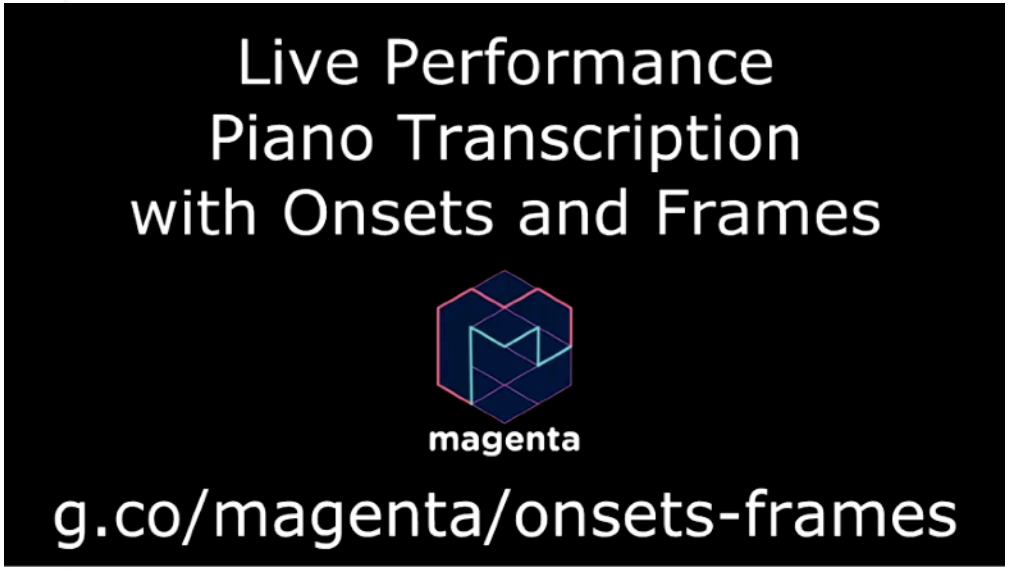
Music has both a vertical and horizontal organisation

- **Vertical** organisation:
  - sources, timbre, polyphony of pitches, chords, key
- **Horizontal** organisation:
  - structure, bar/measure, rhythm

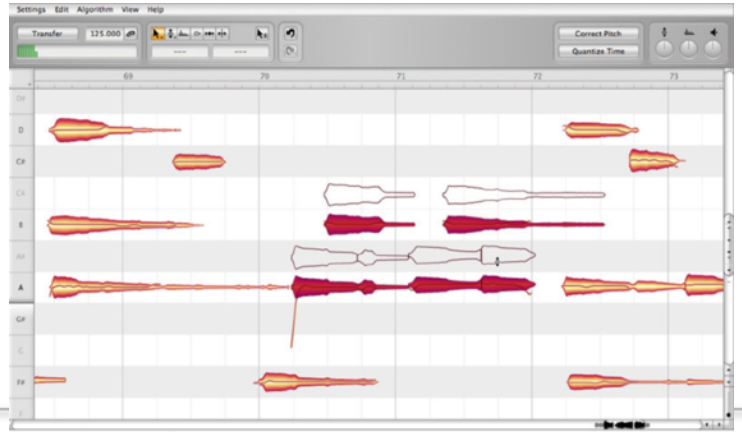


Content-description → pitch





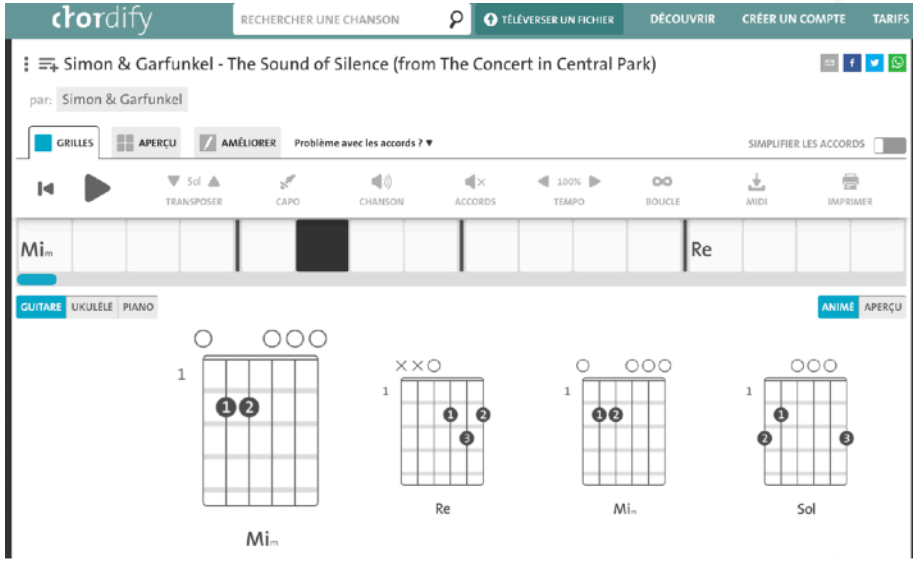
Notes-manipulation



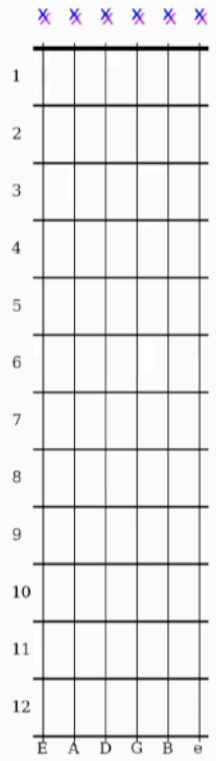
Content-description → chords

Chord transcription

- https://chordify.net/



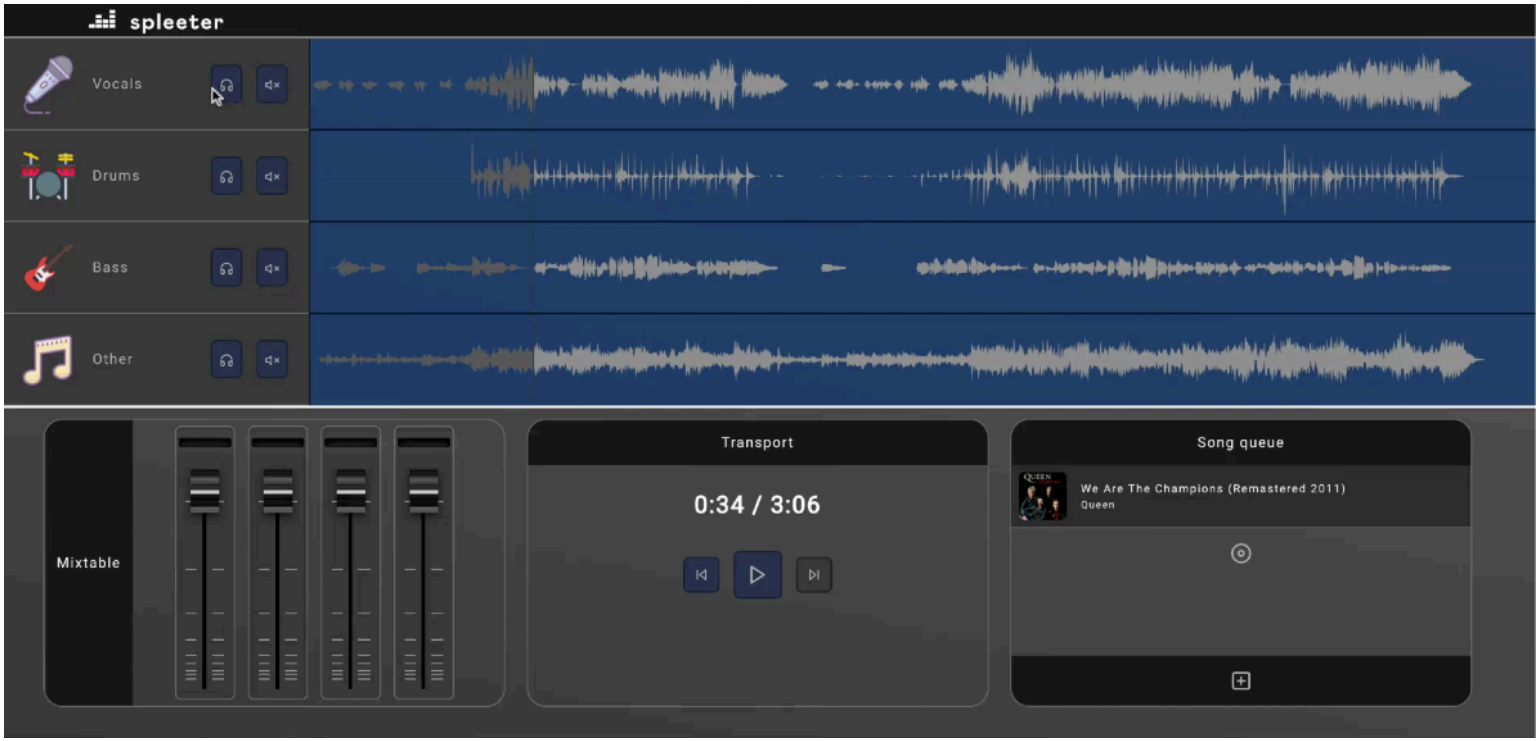
Guitar tab transcription



[A. Wiggins and Y. Kim. "**Guitar tablature estimation with a convolutional neural network.**", ISMIR 2019]

Music source separation

- Spleeter by Deezer



https://github.com/deezer/spleeter

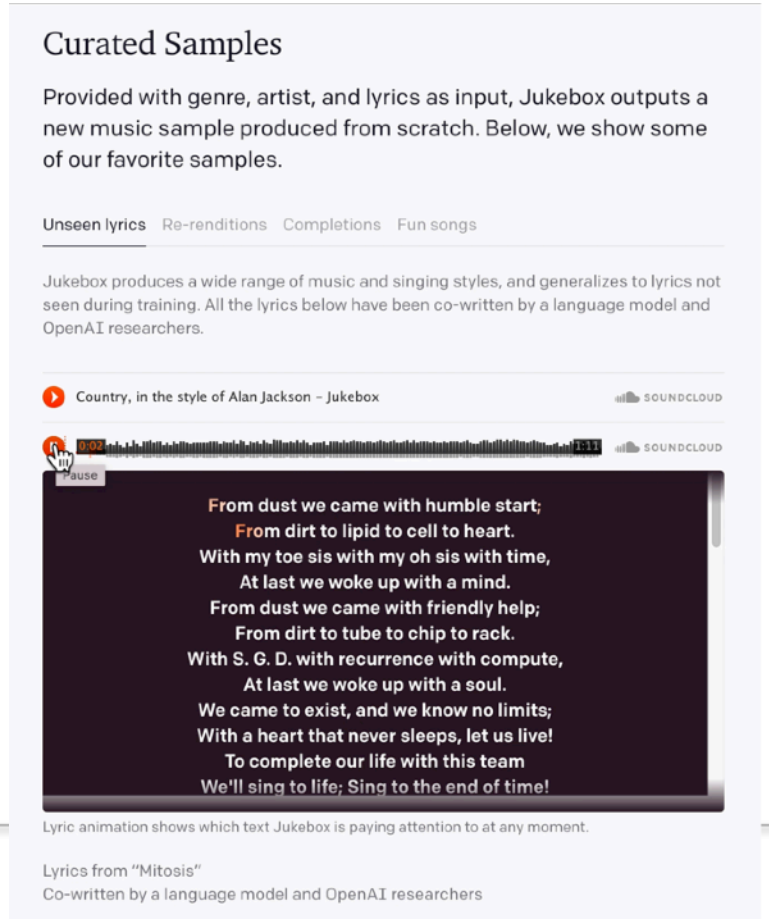
Music Generation

magenta

https://magenta.tensorflow.org/

OpenAI

https://openai.com/blog/jukebox/



Various categories of audio content
Applications

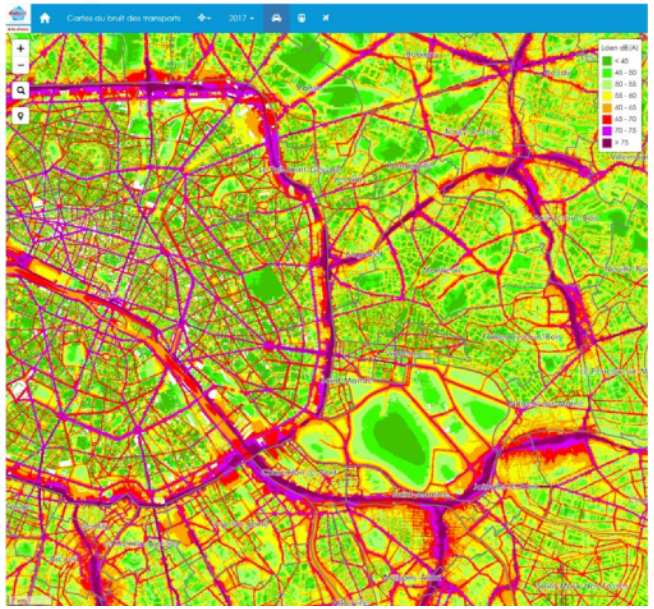
Description	**Envirronmental Sounds** **Acoustic scene classificaton** **Sound event localization, detection**
Transformation	
Generation	**Sound texture synthesis**
	DCASE http://dcase.community

Environmental Sounds Applications

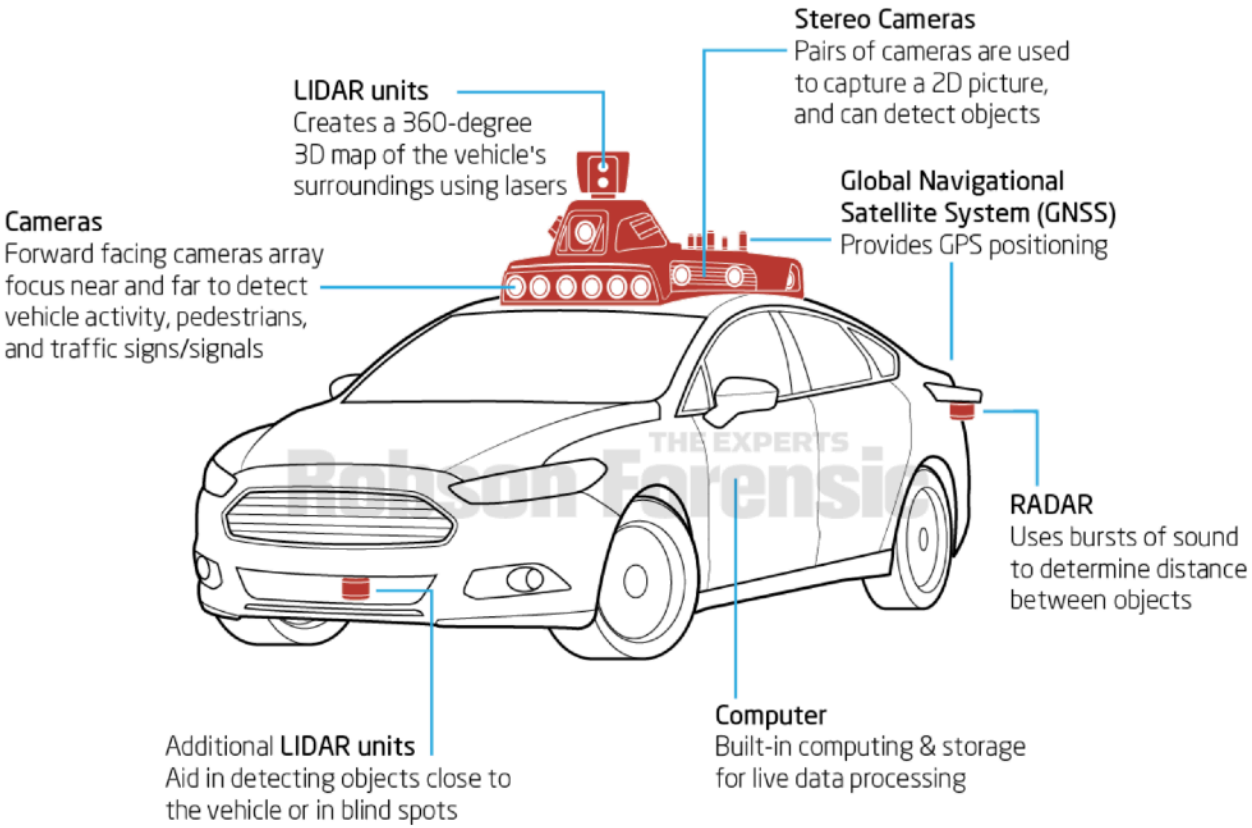
Smart cities

- Sonyc : https://wp.nyu.edu/sonyc/
- Birdvox : https://wp.nyu.edu/birdvox/

BRUITPARIF

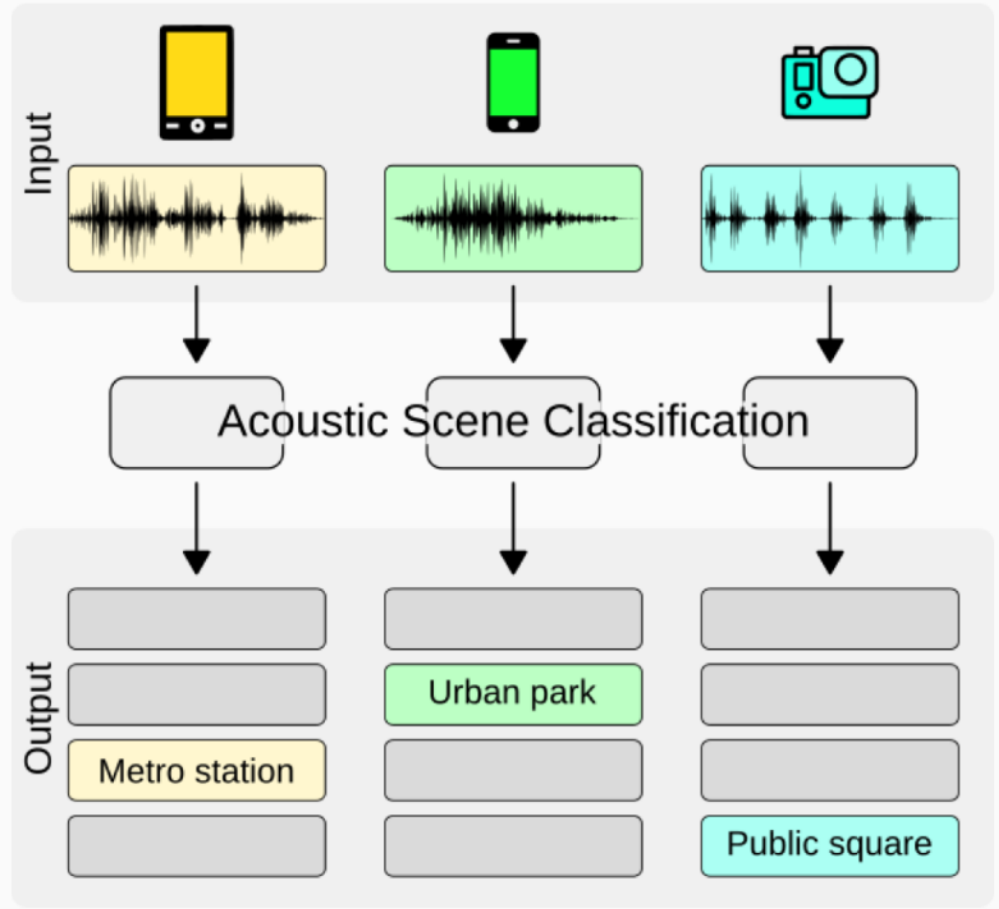


Autonomous agents



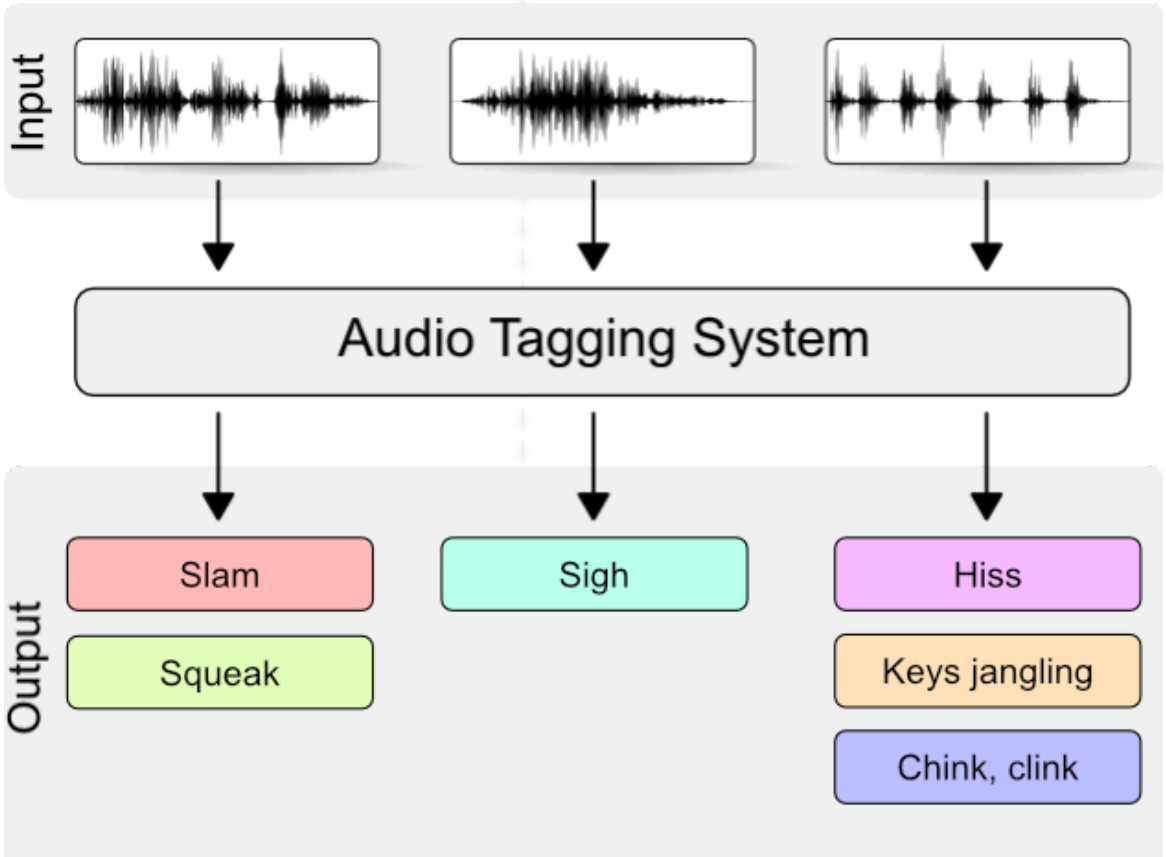
Task 1

- Acoustic Scene Classification



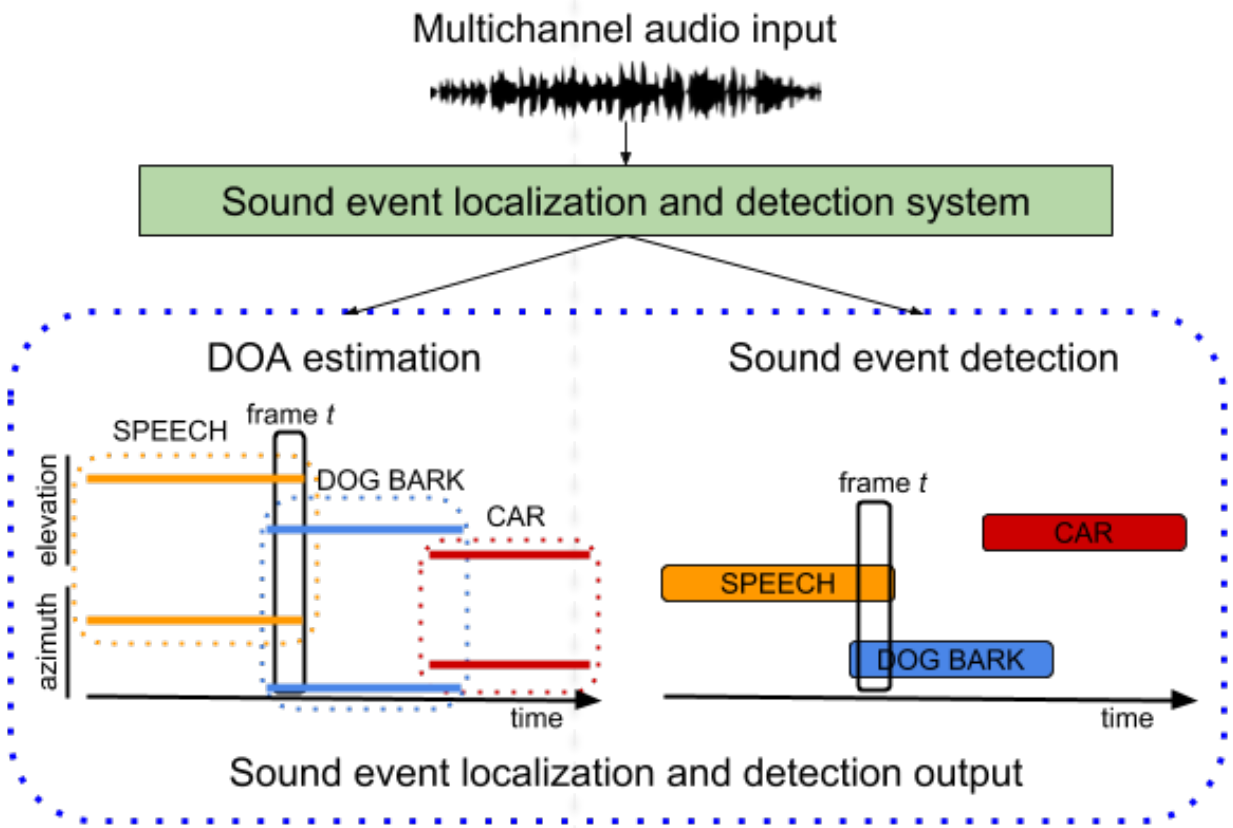
Task 2

– Audio tagging with noisy labels and minimal supervision



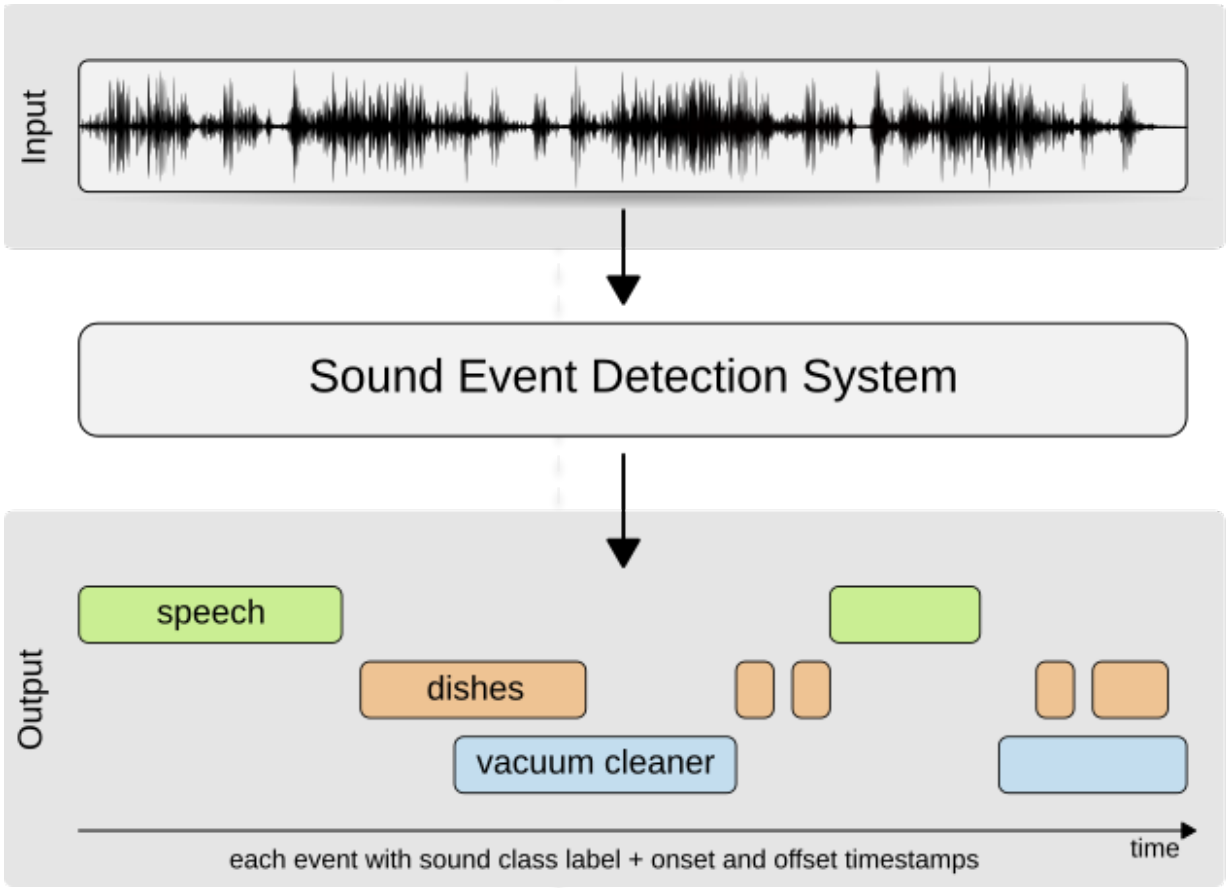
Task 3

– Sound Event Localisation and Detection



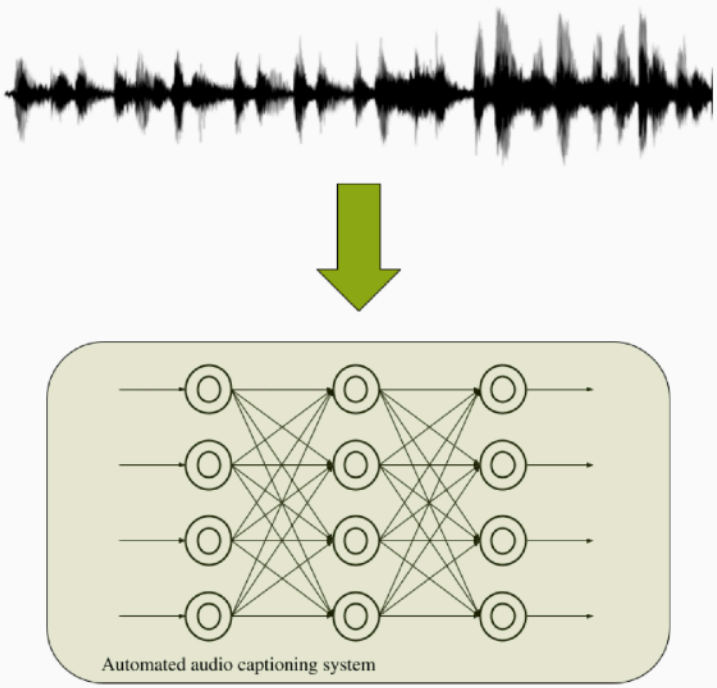
Task 4

– Sound event detection in domestic environments



Task 6

– Automated audio captioning (AAC)



Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

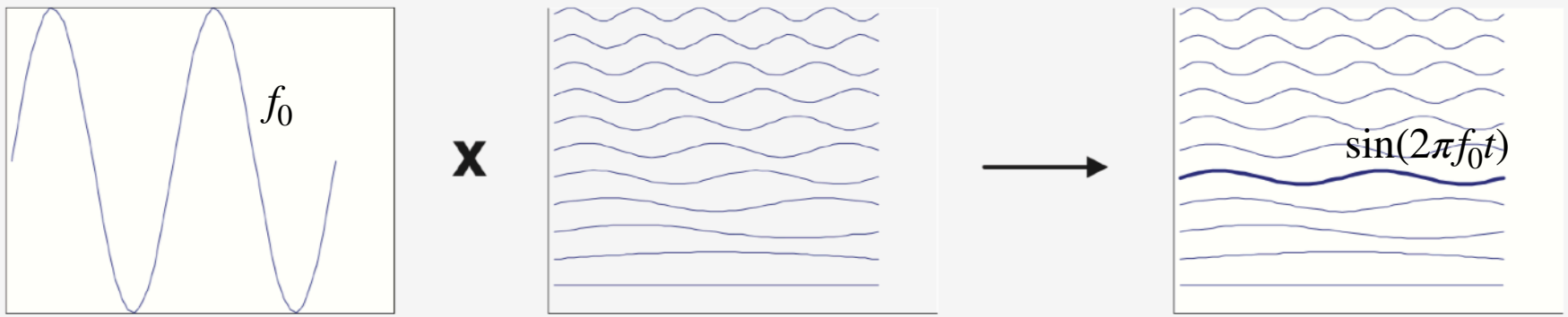
Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

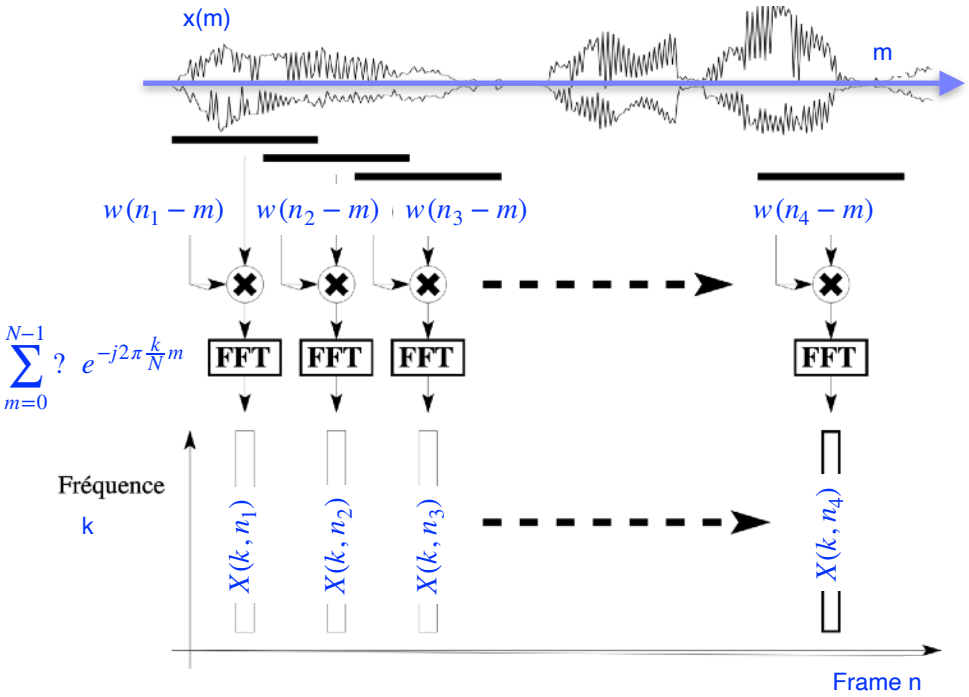
Fourier Transform
continuous time and frequency

- $$X(\omega) = \int_t x(t) e^{-j\omega t} dt \quad X(f) = \int_t x(t) e^{-j(2\pi f)t} dt$$
  - Variables
    - t : time
    - $\omega = 2\pi f$: continuous frequency in radian
    - $\exp(j2\pi ft) = \cos(2\pi ft) + j\sin(2\pi ft)$
  - Why the Fourier Transform ?
    - It is difficult to interpret directly the value of the waveform $x(t)$
    - Reproduce the frequency decomposition of human earing



Fourier Transform short time (STFT)

- $$X(k,n) = \sum_{m=-\infty}^{+\infty} \underline{w(n-m)}\, x(m)\, e^{-j2\pi\frac{k}{N}m} \quad \forall k \in [0, N-1]$$
  - Apply the DFT on successive segments of the audio signal,
    - each segment is centred/start around time n
  - Why the STFT ?
    - Audio signal= non-stationary
      - its properties evolve over time
    - "Local" (in time) stationarity
      - over a time duration of $\pm 40\ ms$
    - STFT: a set of successive DFT over segments of $\pm 40\ ms$ duration
      - = Short-time analysis ("frames" in video)



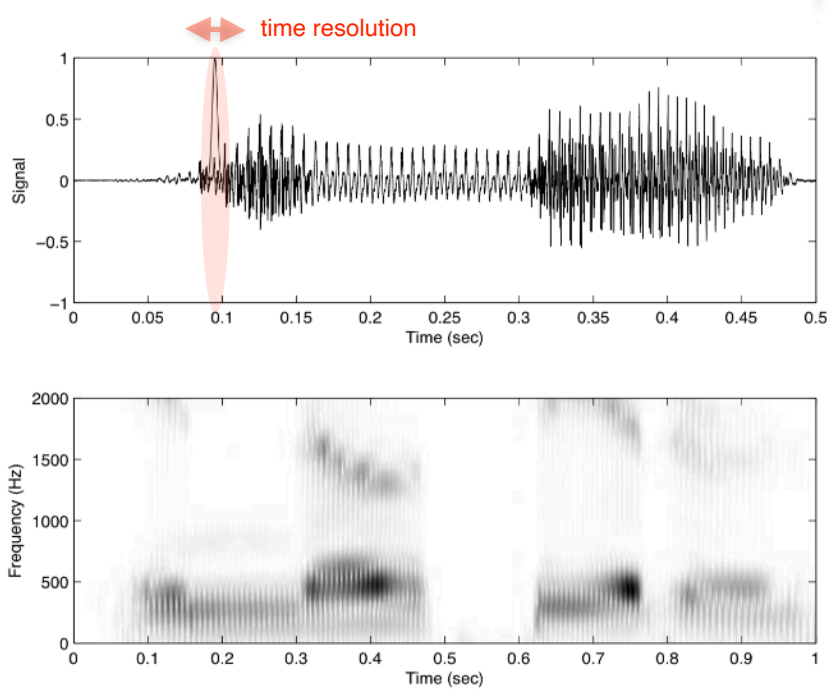
Fourier Transform short time (STFT)

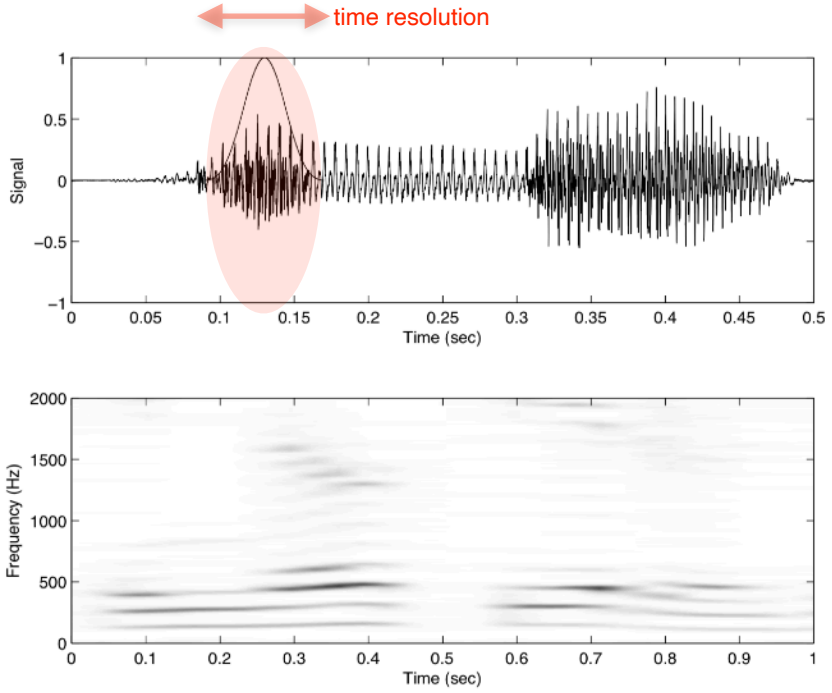
- $$X(k,n) = \sum_{m=-\infty}^{+\infty} \underline{w(n-m)}\, x(m)\, e^{-j2\pi \frac{k}{N} m} \quad \forall k \in [0, N-1]$$
  - Analysis window $w(t)$
  - $x(t) \cdot w(t) \rightleftarrows X(f) \circledast W(f)$
    - $w(t)$ is named "analysis window"
    - Parameters:
      - 1) **type** of the analysis window $w(t)$
      - 2) **time duration** L of the window $w(t)$
    - The type and time duration defines the spectral characteristics of the STFT
      - Width of the main lobe (at $-6dB_{20}$): $Bw = \frac{Cw}{L}$
      - Height of secondary lobes

Propriétés	$x(t)$	$X(f)$
Similitude	$x(at)$	$\frac{1}{\lvert a \rvert} X(\frac{f}{a})$
Linéarité	$ax(t) + by(t)$	$aX(f) + bY(f)$
Translation	$x(t - t_0)$	$X(f) \exp(-j2\pi f t_0)$
Modulation	$x(t) \exp(j2\pi f_0 t)$	$X(f - f0)$
Convolution	$x(t) \circledast y(t)$	$X(f)Y(f)$
Produit	$x(t)y(t)$	$X(f) \circledast Y(f)$
Parité	réelle paire	réelle paire
	réelle impaire	imaginaire paire
	imaginaire paire	imaginaire paire
	imaginaire impaire	réelle impaire
	complexe paire	complexe paire
	complexe impaire	complexe impaire
	réelle	$X(f) = X^*(-f)$
		$\Re(X(f))$ est paire
		$\Im(X(f))$ est impaire
	$x^*(t)$	$X^*(f)$

Fourier Transform short time (STFT)

- **Time and frequency paradox**
  - It is not possible to observe the signal with a good time and frequency localisation simultaneously !

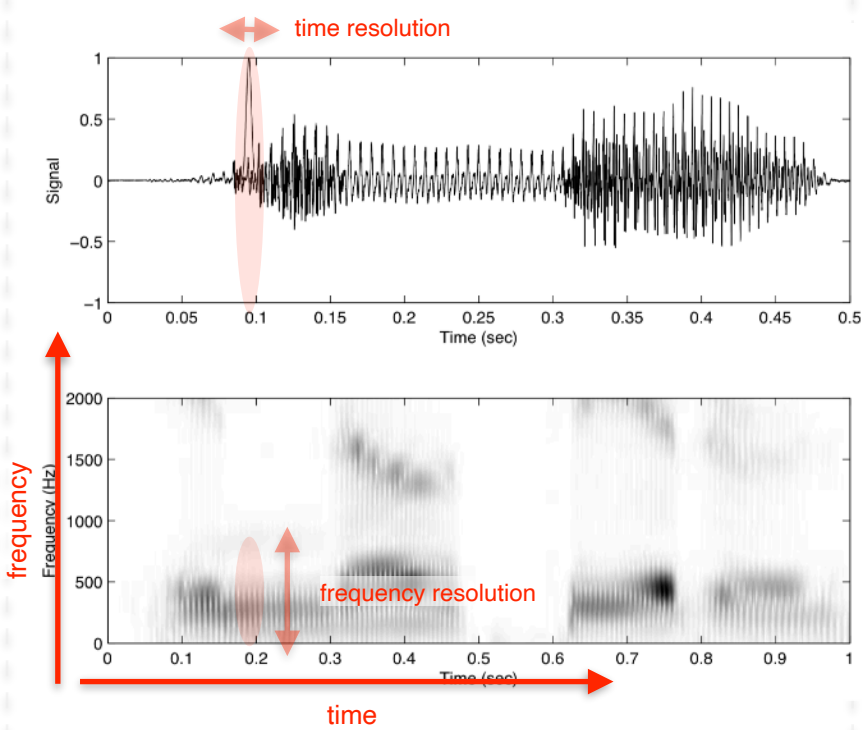


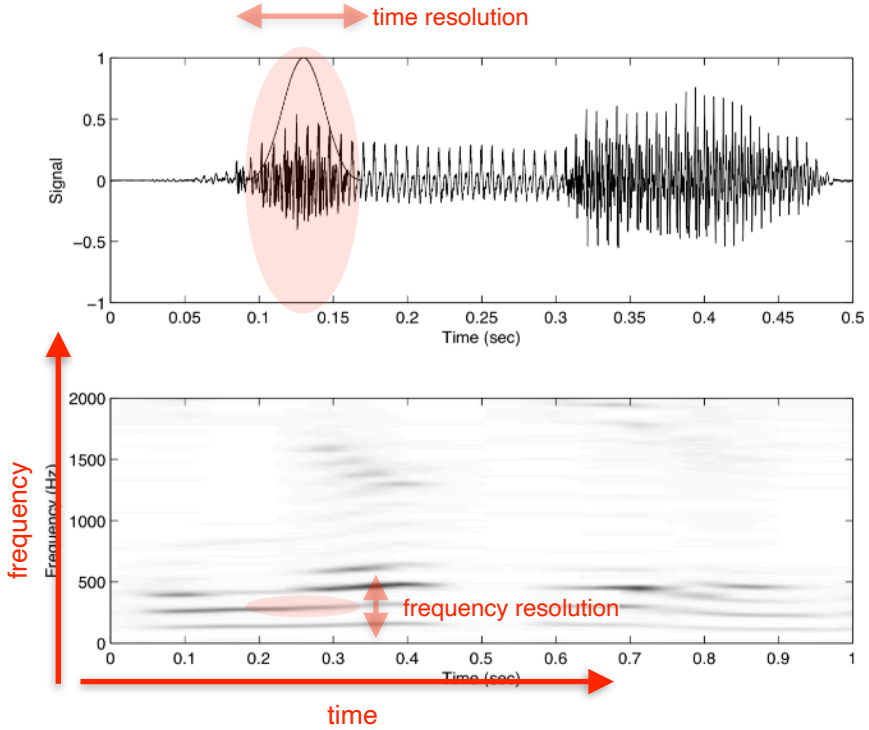


- **How to solve this issue ?**
  - Other transforms: wavelet, constant-Q-transform

Fourier Transform short time (STFT)

- **Time and frequency paradox**
  - It is not possible to observe the signal with a good time and frequency localisation simultaneously !

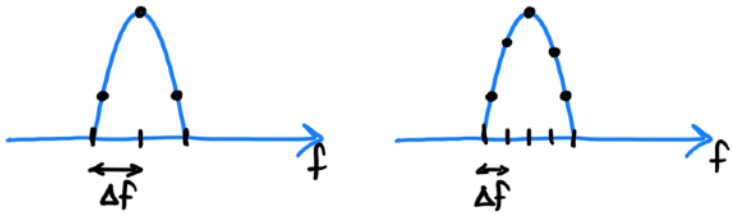




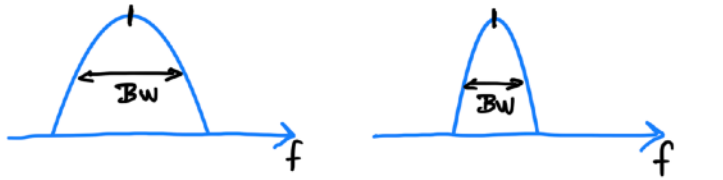
- **How to solve this issue ?**
  - Other transforms: wavelet, constant-Q-transform

Constant-Q-Transform (CQT)

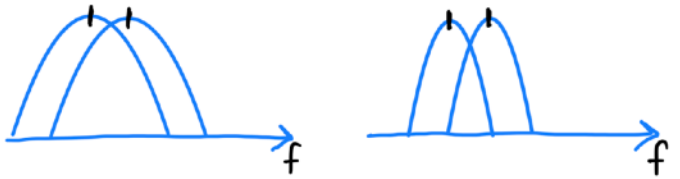
- Discrete Fourier Transform (DFT)
  - _ Definition : Spectral **precision** : $\Delta f = \frac{sr}{N}$
    - it is the step-size at which the Fourier spectrum is sampled
    - it depends on the size of the DFT: N
    - we can improve the precision by increasing N



  - _ Definition : Spectral **resolution** : $B_w = \frac{C_w}{L}$
    - it describes the ability to discriminate (separate in the spectrum) two adjacent simultaneous frequencies



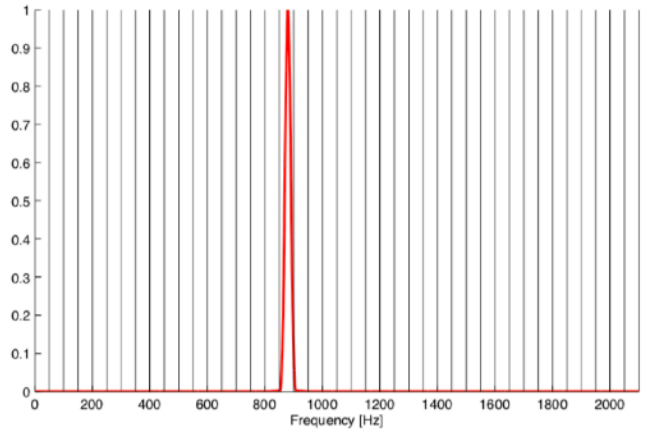
  - – Warning :
    - even if we increase N (zero-padding) while keeping L constant will not improve the resolution !



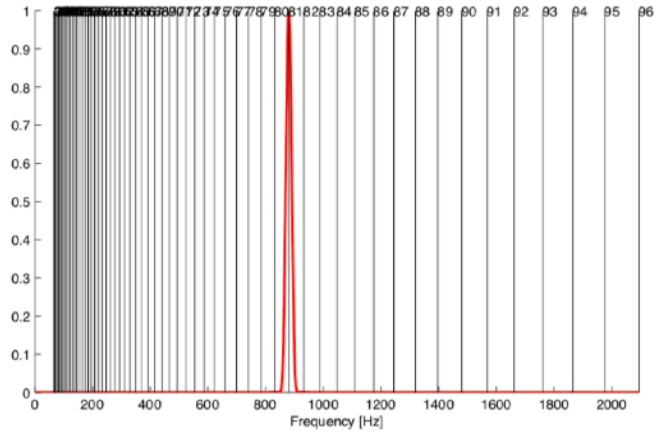
- In a DFT:
  - – Spectral precision and resolution are constant over frequencies

Constant-Q-Transform (CQT)

- In musical audio
  - the frequencies of the pitches are logarithmically spaced $f_k = f_0 \cdot 2^{\frac{k}{12}}$
    - if we choose A-4 = la-3 = 440 Hz as the reference
    - to go from midi-pitches m_k to frequencies f_k:
      - $f_k = 440 \cdot 2^{\frac{m_k - 69}{12}}$
    - to go from frequencies f_k to midi-pitches m_k:
      - $m_k = 12 \log_2 \frac{f_k}{440} + 69$
  - pitch frequencies are
    - close together in low frequencies,
    - distant in high frequencies
  - The **spectral resolution** of the DFT
    - is not sufficient (to separate adjacent notes) in low frequencies
    - is too large for high frequencies



Espacement linéaire de la DFT



Espacement logarithmique des hauteurs de notes

Constant-Q-Transform (CQT)

- Solution ?
  - Change the spectral resolution B_w depending on the frequency f_k being considered

- How ?
  - By changing the window length L for eadh frequency f_k
  - The factor $Q = \frac{f_k}{f_{k+1} - f_k}$ should remains constant in frequency
    - $Q = \frac{f_k}{Bw} = \frac{f_k}{Cw/L} = \frac{f_k \cdot L}{Cw}$
  - We choose a different L for each frequency f_k
    - $L_k = \frac{Q \cdot Cw}{f_k}$


An efficient algorithm for the calculation of a constant Q transform

Judith C. Brown

Physics Department, Wellesley College, Wellesley, Massachusetts 02181 and Media Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

Miller S. Puckette

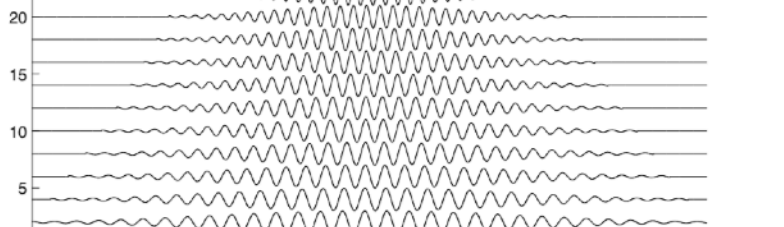
IRCAM, 31 rue St Merri, Paris 75004, France

(Received 5 February 1992; revised 10 April 1992; accepted 16 June 1992)

An efficient method of transforming a discrete Fourier transform (DFT) into a constant Q transform, where Q is the ratio of center frequency to bandwidth, has been divided. This method involves the calculation of kernels that are then applied to each subsequent DFT. Only a few multiplies are involved in the calculation of each component of the constant Q transform, so this transformation adds a small amount to the computation. In effect, this method makes it possible to take full advantage of the computational efficiency of the fast Fourier transform (FFT). Graphical examples of the application of this calculation to musical signals are given for sounds produced by a clarinet and a violin.

PACS numbers: 43.60.Gk, 43.75.Yy, 43.75.De, 43.75.Zz

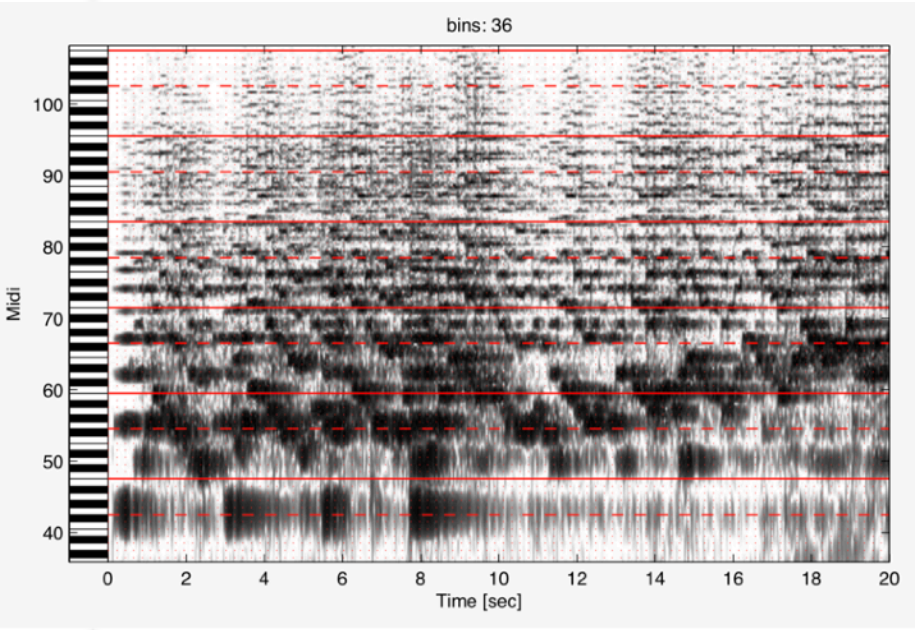
I. THEORY



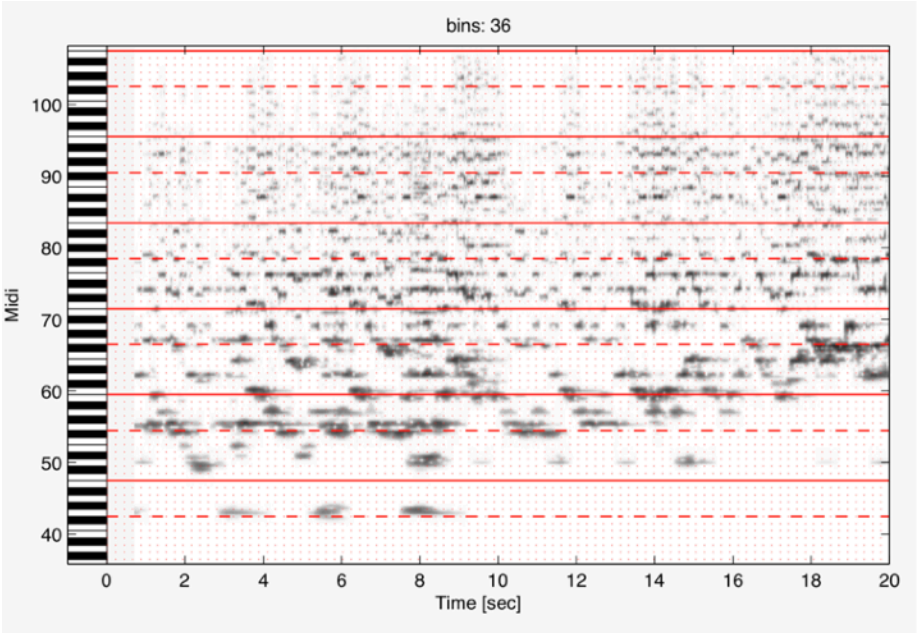
[J. Brown and M. Puckette. An efficient algorithm for the calculation of a constant q transform. JASA, 1992.]

Constant-Q-Transform (CQT)

Example (using DFT)

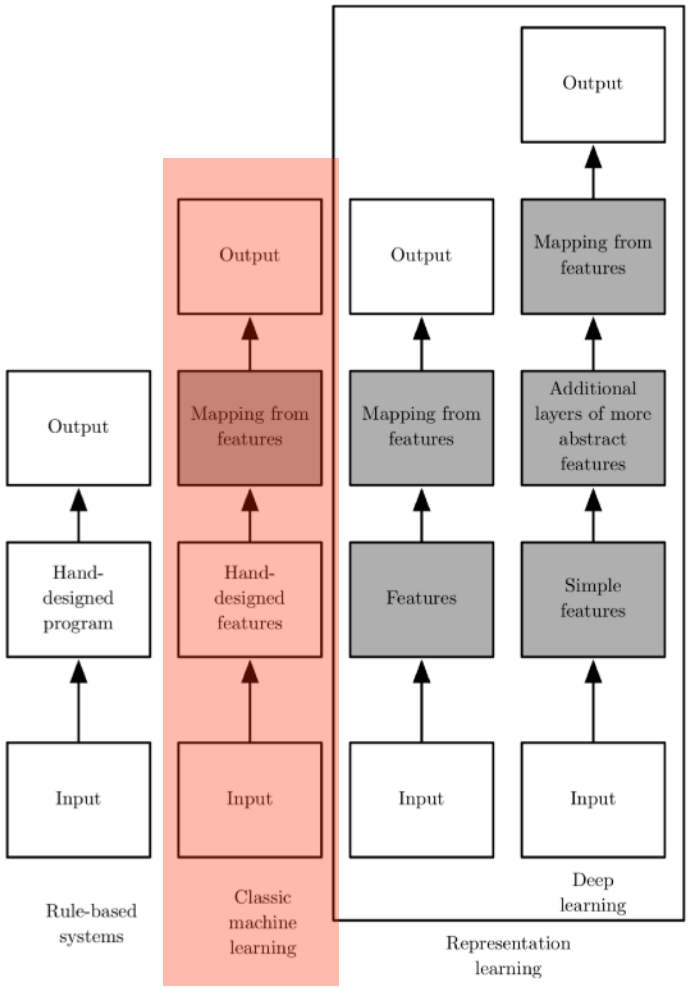


Example (using the CQT)



Traditional machine-learning approach

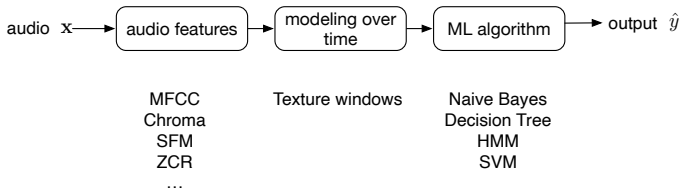
Traditional machine-learning approach



I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning. MIT Press, 2016.

Traditional machine-learning approach

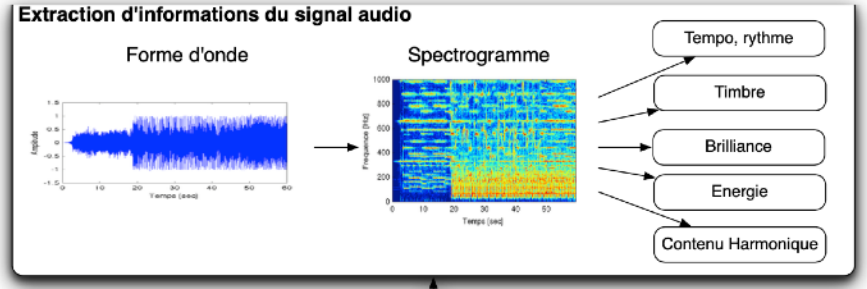
Traditional approach

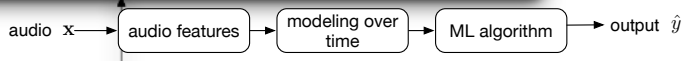


Traditional machine-learning approach

Traditional approach

- **Training phase**
- Testing phase

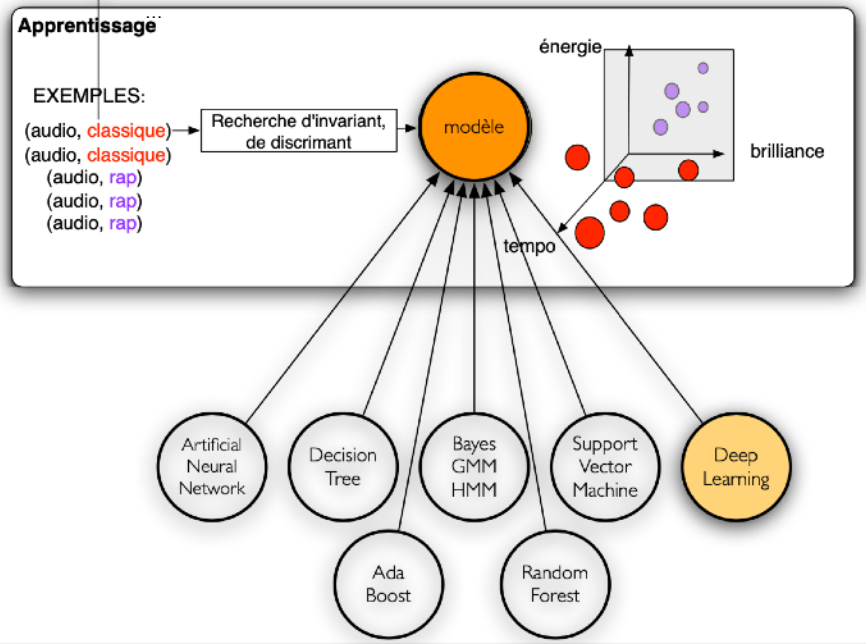


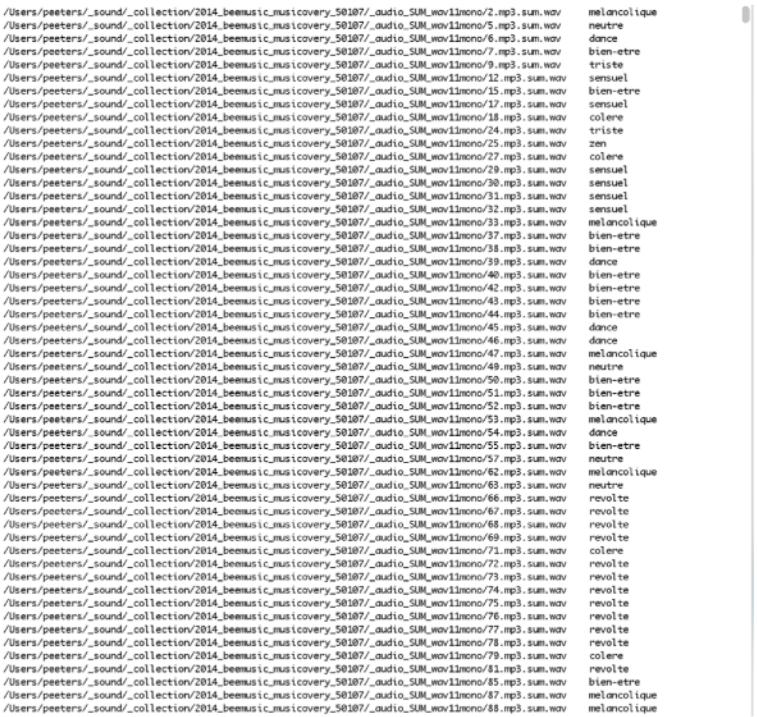


MFCC
Chroma
SFM
ZCR

Texture windows

Naive Bayes
Decision Tree
HMM
SVM



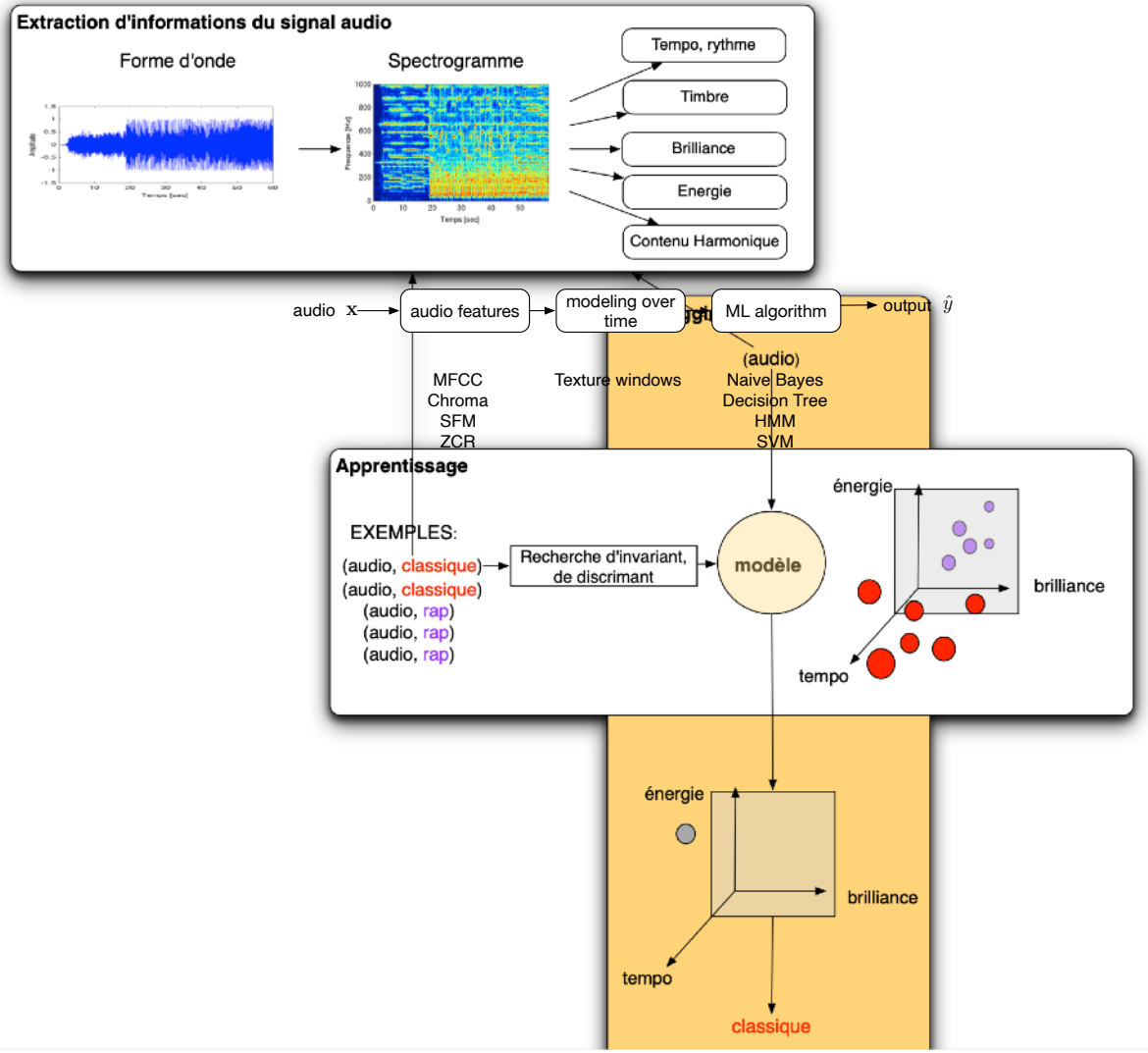


```
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/2.mp3.sum.wav      melancolique
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/5.mp3.sum.wav      neutre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/6.mp3.sum.wav      dance
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/7.mp3.sum.wav      bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/9.mp3.sum.wav      triste
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/12.mp3.sum.wav     sensuel
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/15.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/17.mp3.sum.wav     sensuel
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/18.mp3.sum.wav     colere
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/24.mp3.sum.wav     triste
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/25.mp3.sum.wav     zen
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/27.mp3.sum.wav     colere
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/29.mp3.sum.wav     sensuel
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/30.mp3.sum.wav     sensuel
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/31.mp3.sum.wav     sensuel
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/32.mp3.sum.wav     sensuel
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/33.mp3.sum.wav     melancolique
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/37.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/38.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/39.mp3.sum.wav     dance
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/40.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/42.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/43.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/44.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/45.mp3.sum.wav     dance
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/46.mp3.sum.wav     dance
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/47.mp3.sum.wav     melancolique
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/49.mp3.sum.wav     neutre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/50.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/51.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/52.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/53.mp3.sum.wav     melancolique
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/54.mp3.sum.wav     dance
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/55.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/57.mp3.sum.wav     neutre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/62.mp3.sum.wav     melancolique
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/63.mp3.sum.wav     neutre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/66.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/67.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/68.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/69.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/71.mp3.sum.wav     colere
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/72.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/73.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/74.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/75.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/76.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/77.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/78.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/79.mp3.sum.wav     colere
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/81.mp3.sum.wav     revolte
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/85.mp3.sum.wav     bien-etre
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/87.mp3.sum.wav     melancolique
/Users/peeters/_sound/_collection/2014_beemusic_musicovery_50107/_audio_SUM_wav11mono/88.mp3.sum.wav     melancolique
```

Traditional machine-learning approach

Traditional approach

- Training phase
- **Testing phase**



Audio features

Audio features examples

Zero-crossing rate (zcr)

- Measures the number of times the audio waveform cross the zero-axis
  - $$zcr = \frac{1}{N}\sum_{n=1}^{N} |sign(x_n) - sign(x_{n-1})|$$
- Usage: allows to distinguish
  - harmonic sounds → low zcr
  - noise sounds→ high zcr

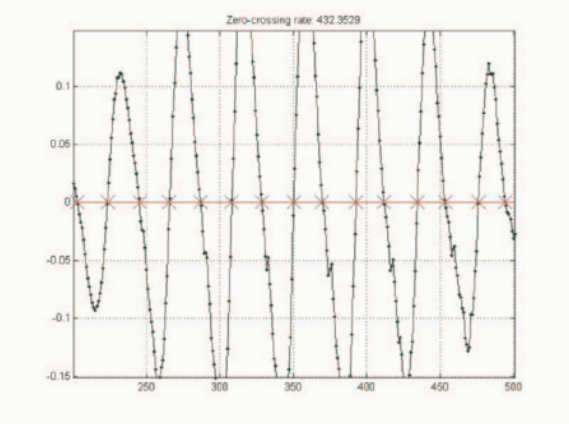


Figure 12 Zero-crossing rate (=432) during voiced speech region

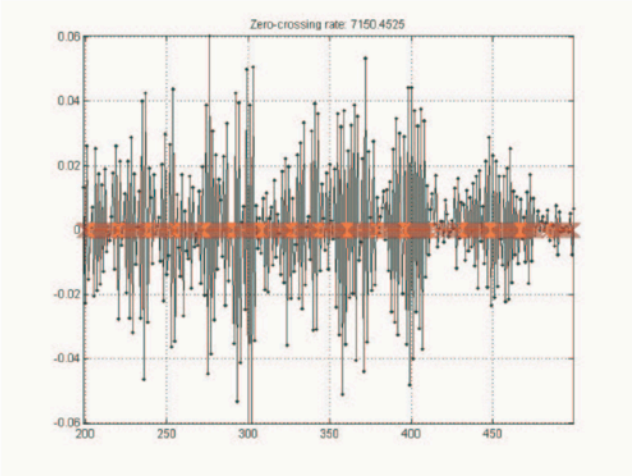
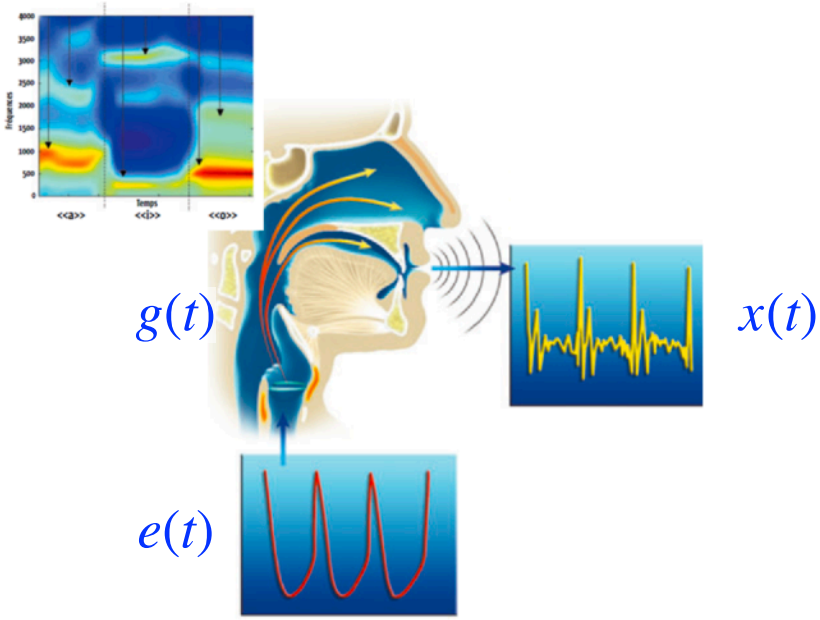


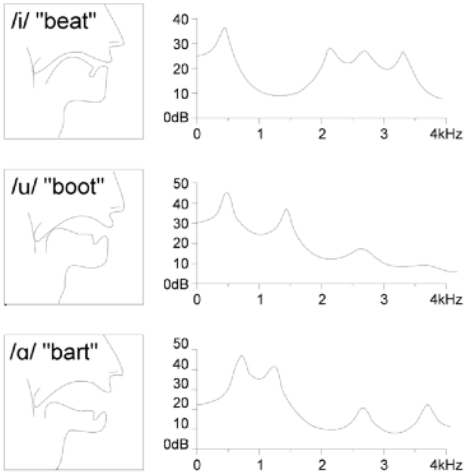
Figure 13 Zero-crossing rate (=7150) during unvoiced speech region

Source/Filter model

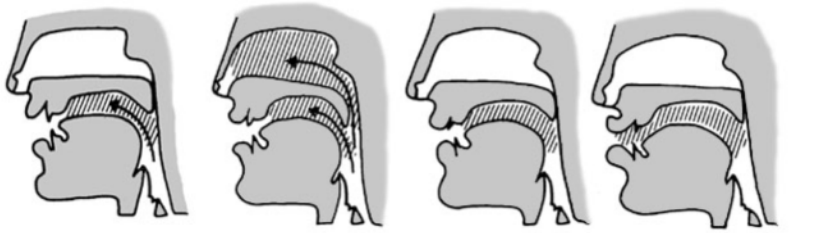
Model:

- represent the signal $x(t)$ as the results of a periodic pulse signal convolved with a resonant filters
- Example:
  - speech (voiced part)
    - vocal folds $e(t)$
    - mouth (resonance), nose (anti-resonance) $g(t)$
  - many musical instruments (trumpet)





source : Mike Brookes



source : outilsrecherche.over-blog.com

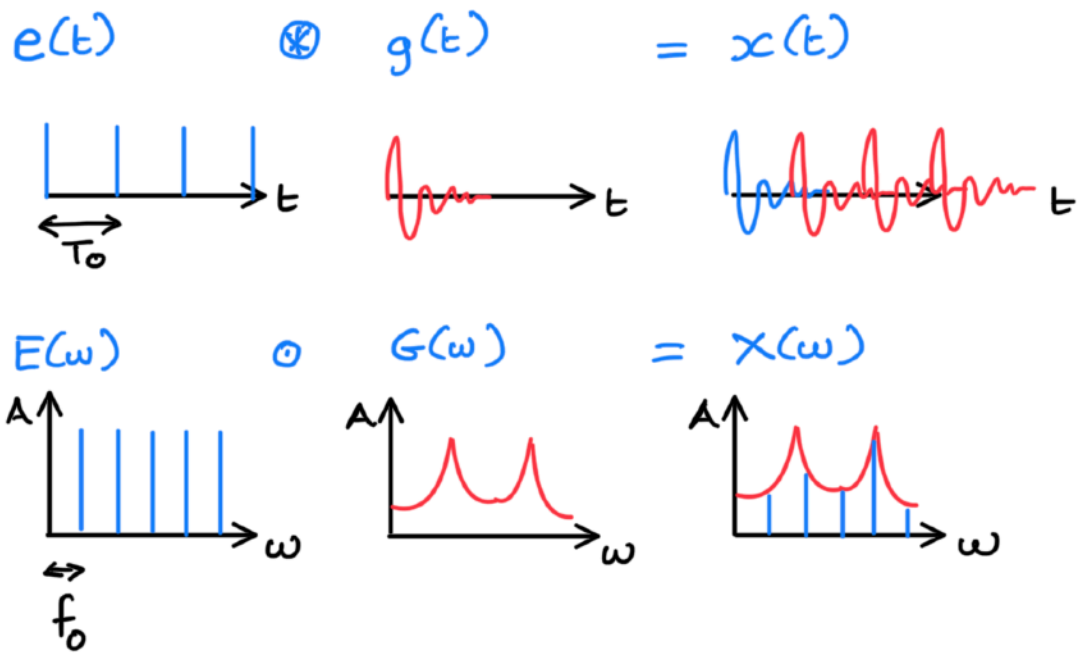
Source/Filter model

In time:

$$x(t) = e(t) \circledast g(t)$$

In frequency:

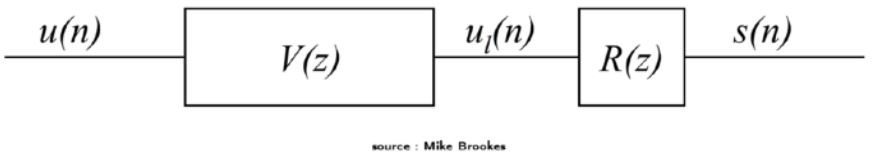
$$X(\omega) = E(\omega) \cdot G(\omega)$$



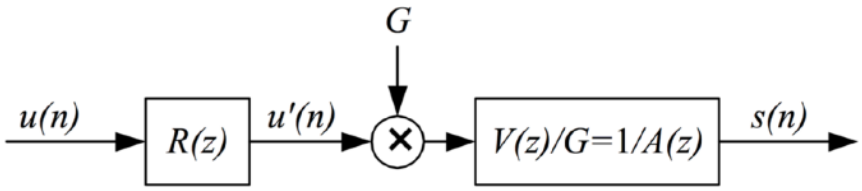
Source/Filter model

Resonant filter:

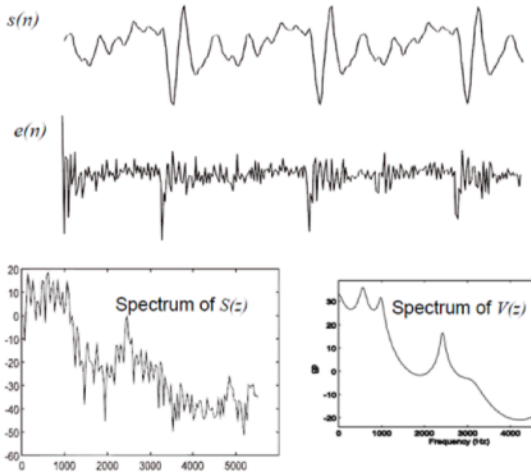
- $V(z) = \dfrac{G}{1 - \sum_{j=1}^{P} a_j z^{-j}}$
  - $x(n) = Gu'(n) + \sum_{j=1}^{P} a_j x(n-j)$
- If the gain of the resonances of the vocal fold are import, the second term dominates and
  - $x(n) \simeq \sum_{j=1}^{P} a_j x(n-j)$
  - This is an **auto-regressive model**



source : Mike Brookes



source : Mike Brookes



Audio features examples
Mel Frequency Cepstral Coefficients (1)

Complex cepstrum

– **Goal**

- describe the shape of the spectrum (the timbre) of a signal using a reduced set of coefficients

– **Complex cepstrum** $c(\tau)$

$$c(\tau) = TF^{-1}\left[\log(X(\omega))\right]$$
$$= \frac{1}{2\pi}\int_{\omega} \log[X(\omega)] \cdot e^{j\omega\tau} d\omega$$

- τ is named "**quefrency**" (=frequency in reverse order)
- $x(t) \xrightarrow{TF} X(\omega) \xrightarrow{\log} \log(X(\omega)) \xrightarrow{TF^{-1}} c(\tau)$

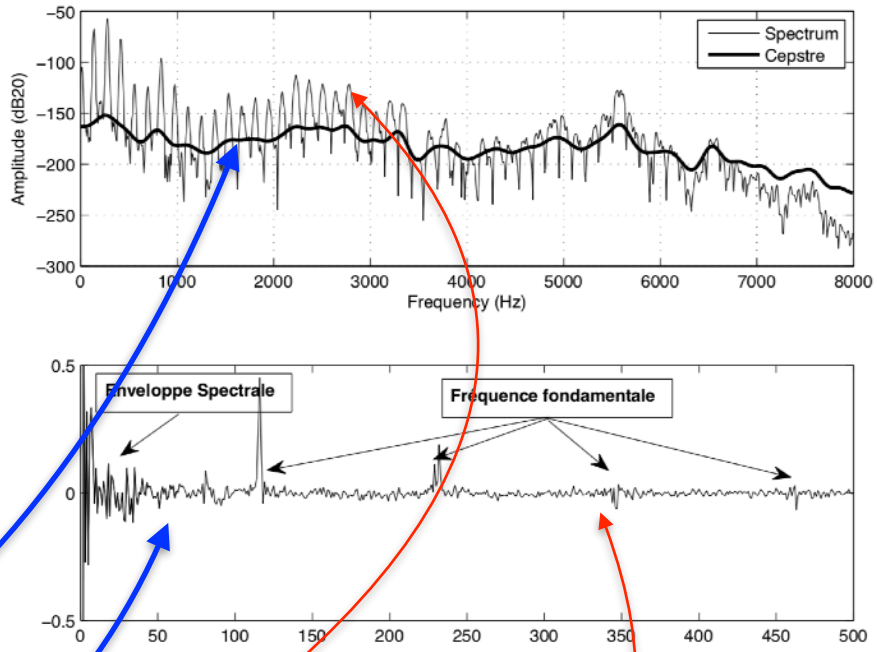
Audio features examples
Mel Frequency Cepstral Coefficients (2)

Source/filter model

- Source $e(t)$: periodic signal
- Filter $g(t)$: resonant/ anti-resonant filter

$$x(t) = e(t) \circledast g(t)$$

$$\xrightarrow{TF} X(\omega) = E(\omega) \cdot G(\omega)$$



$$\xrightarrow{\log} \log(X(\omega)) = \underbrace{\log[G(\omega)]}_{\text{slow variations over } \omega} + \underbrace{\log[E(\omega)]}_{\text{fast variations over } \omega}$$

$$\xrightarrow{TF^{-1}} TF^{-1}\left[\log(X(\omega))\right] = \underbrace{TF^{-1}\left[\log[G(\omega)]\right]}_{\text{energy at quefrency } \tau<<} + \underbrace{TF^{-1}\left[\log[E(\omega)]\right]}_{\text{energy at quefrency } \tau>>}$$

Audio features examples
Mel Frequency Cepstral Coefficients (4)

Mel Frequency Cepstral Coefficients (MFCCs)

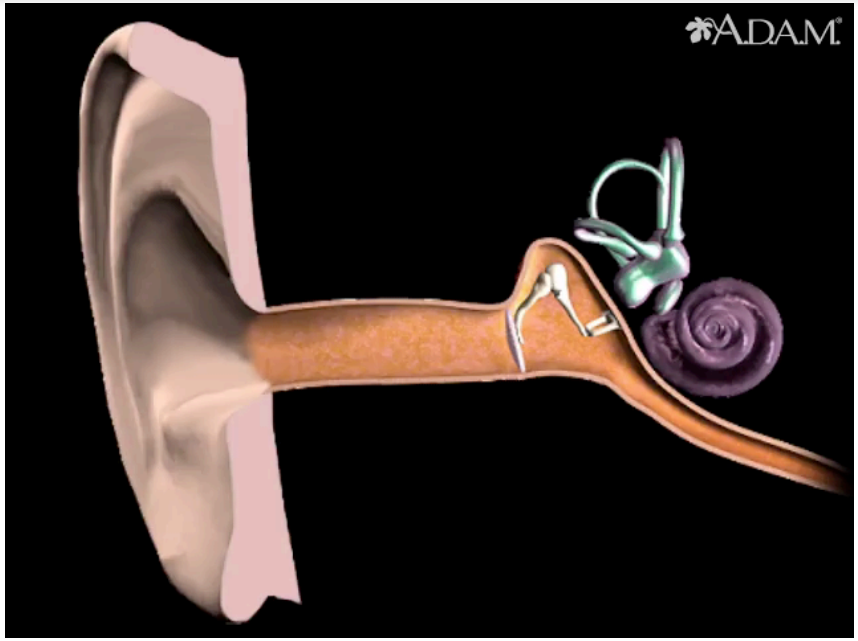
- MFCC ? = real cepstrum computed on the power spectrum $|X(\omega)|^2$ converted to the Mel scale (a perceptual scale)

- **Why perceptual scales ?**
  - Fourier Transform
    - decomposition on a set of sinusoidal components which frequencies are linearly spaced (f_k = 10Hz, 20Hz, 30Hz, ... Hz)
  - Human hearing:
    - decomposition on a set of filters which frequencies are logarithmically spaced (10, 20, 40, 80, ... Hz).
    - highest resolution in low-frequencies, lowest resolution in high frequencies
    - in speech, formants/resonances are closer together in low frequencies
  - MFCCs allows a more compact representation than the real cepstrum
- **How ?**
  - Use of perceptual scales: Mel-scale, Bark-scale, ERB-filters, Gamma-tone filters
- **Usage ?**
  - MFCCs are the most used features in audio: speech, music, environmental sounds recognition, ...

Audio features examples
Mel Frequency Cepstral Coefficients (5)

Human hearing

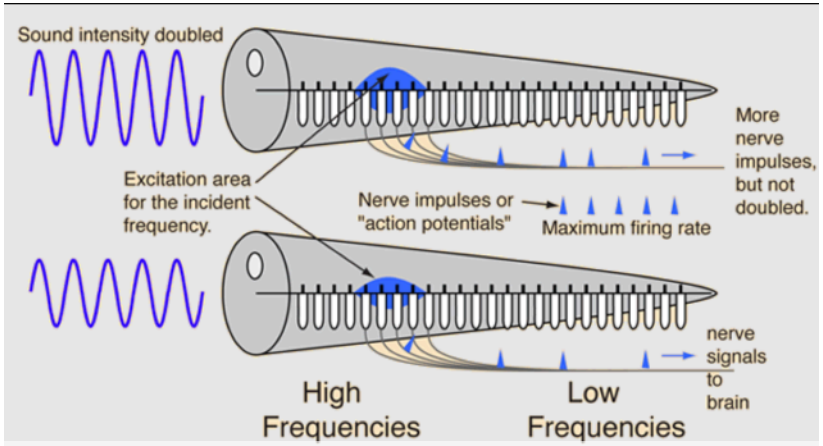
- Cochlea



https://medlineplus.gov/ency/anatomyvideos/000063.htm

- Critical bands
  - perception of two tones at f_1 and f_2
  - perception of a beating-tone at $\frac{f_1+f_2}{2}$

$$\cos f_1 + \cos f_2 = 2\cos\frac{f_1-f_2}{2}\cos\frac{f_1+f_2}{2}$$



source: http://hyperphysics.phy-astr.gsu.edu/hbase/Sound/loud.html

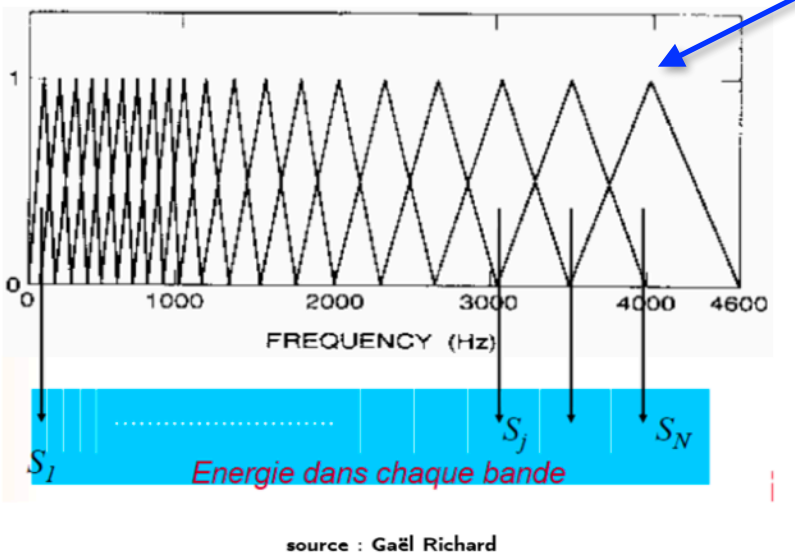
Audio features examples
Mel Frequency Cepstral Coefficients (6)

Mel scale ?

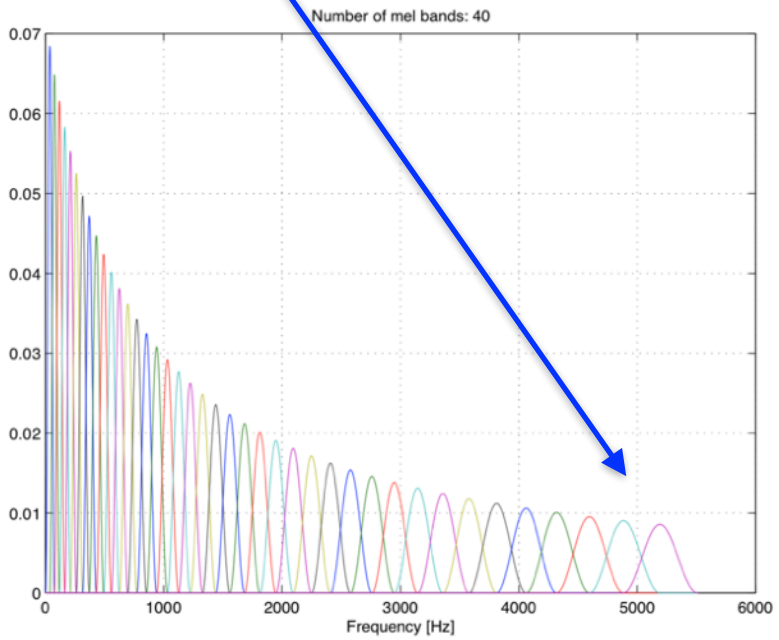
$$mel(f) = \frac{1000}{\ln 2} \ln\left(1 + \frac{f}{1000}\right)$$

- Remark: variations of the constant exist

different shapes for the filter: triangular, hanning, tanh



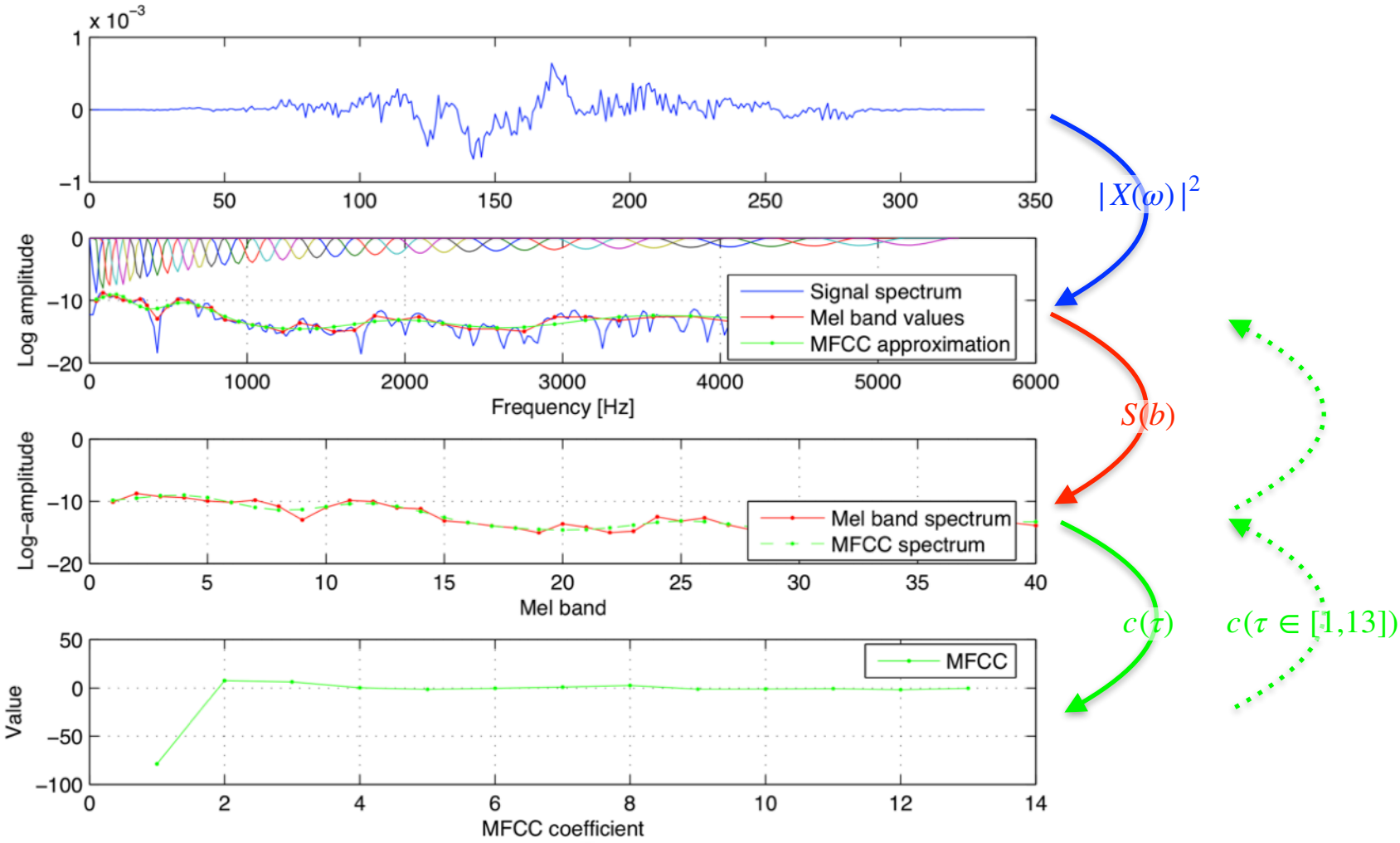
source : Gaël Richard



Fant, Gunnar. (1968). *Analysis and synthesis of speech processes.* In B. Malmberg (Ed.), *Manual of phonetics* (pp. 173-177). Amsterdam: North-Holland.

Audio features examples
Mel Frequency Cepstral Coefficients (8)

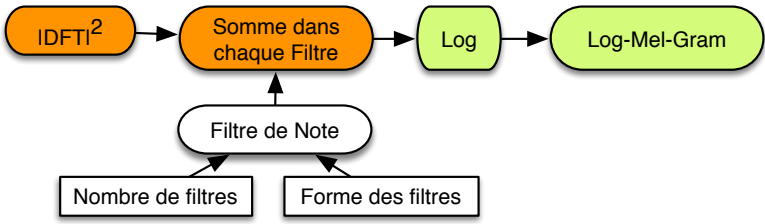
Example of the computation of MFCCs

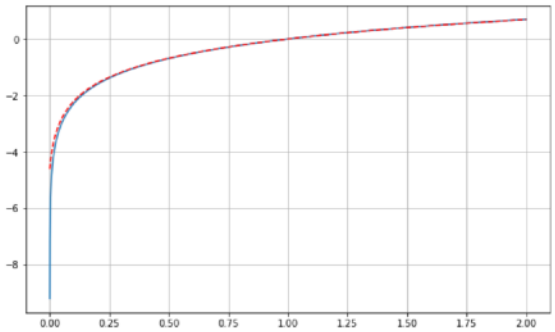


Audio features examples
Mel Frequency Cepstral Coefficients (9)

Variation for the case of DNN inputs: the Log-Mel-gram

- In the **Cepstrum**
  - the DCT is used to separate the contribution of the source and the filter
- In the **Mel Frequency Cepstral Coefficients**
  - the Mel-bands already mostly represent the filter contribution
  - the DCT is mostly used to de-correlated the dimensions
    - (latter used in GMM:diagonal covariance matrix Σ)
- **Log-Mel-Gram**
  - When using DNN, we don't need such a de-correlation of the inputs
  - we then bypass the DCT of the MFCC → Log-Mel-Gram
  - Tricks: to avoid singularity of $\log(x)$ → replace log by $\log(1+\gamma x)$ with $\gamma = 100$





Sinusoidal (harmonic) model

Model:

- represent the signal $x(t)$ as a sum of (harmonic) sinusoidal components + noise

$$x(t) = \sum_{h=1}^{H(t)} A_h(t) \cdot \cos(\phi_h(t)) + b(t)$$

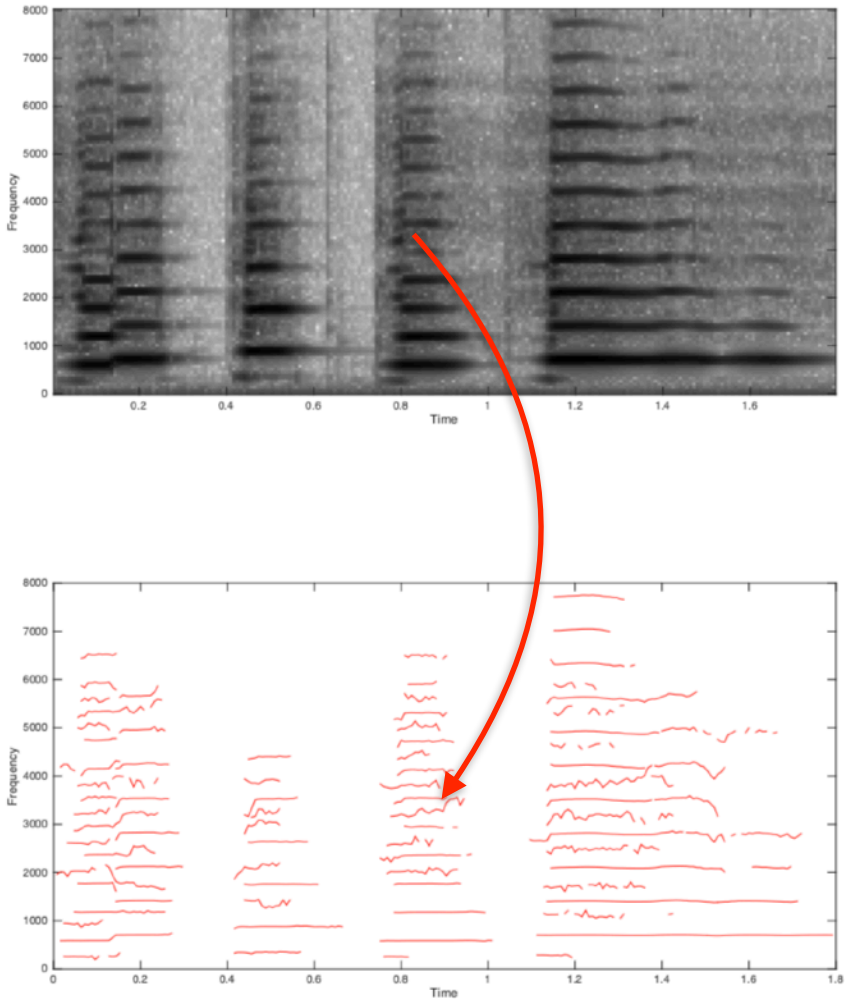
- Fourier series
- Organ musical instrument

Hypothesis:

- Number of components $H(t)$ is reduced ($\neq$ Fourier Transform)
- Parameters slowly vary over time
  - local stationarity
  - we can approximate locally in time
    - $A_h(t) = A_h(t_m)$
    - $\phi(t) = 2\pi f_h(t_m)(t - t_m) + \phi_h(t_m)$

$$\hat{x}_m(t_m) = \sum_{h=1}^{H(t)} A_h(t_m) \cdot \cos\left(2\pi f_h(t_m)(t - t_m) + \phi_h(t_m)\right)$$

$$\hat{x}(t) = \sum_{m} \hat{x}_m(t) + b(t)$$

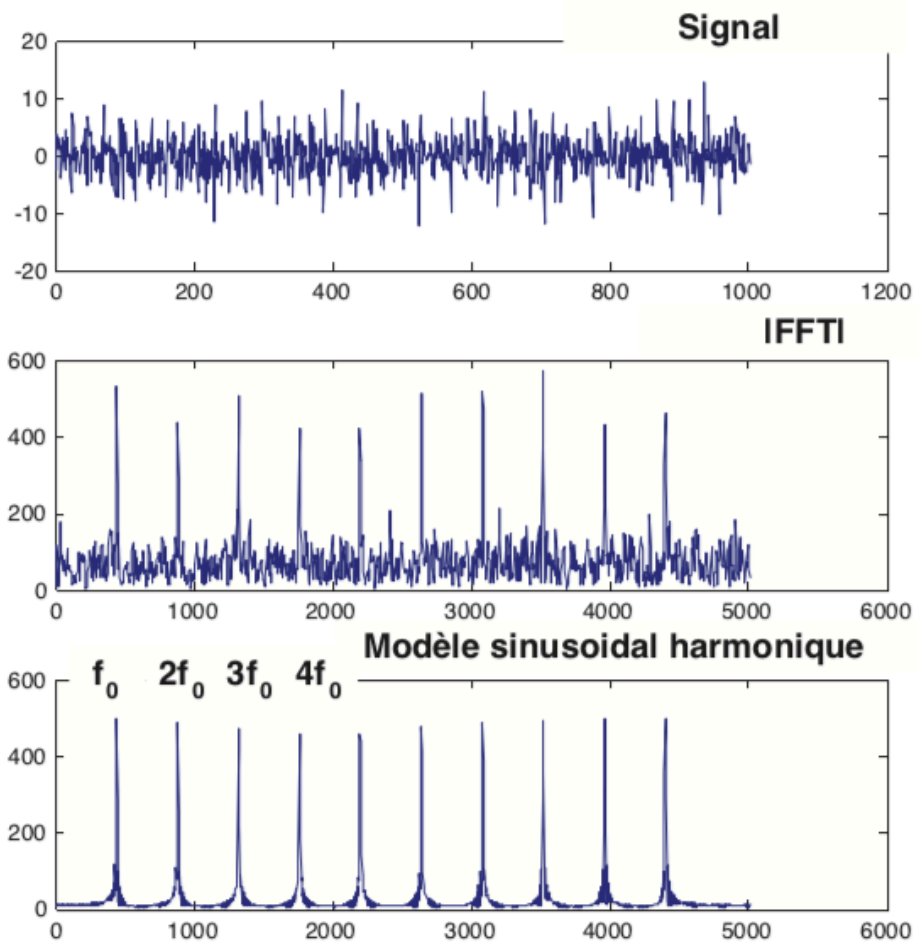


[X. Serra and J. Smith. "**Spectral modeling synthesis: A sound analysis/synthesis system** ..." In CMJ, 1990] LINK

Sinusoidal (harmonic) model

Hypothesis:

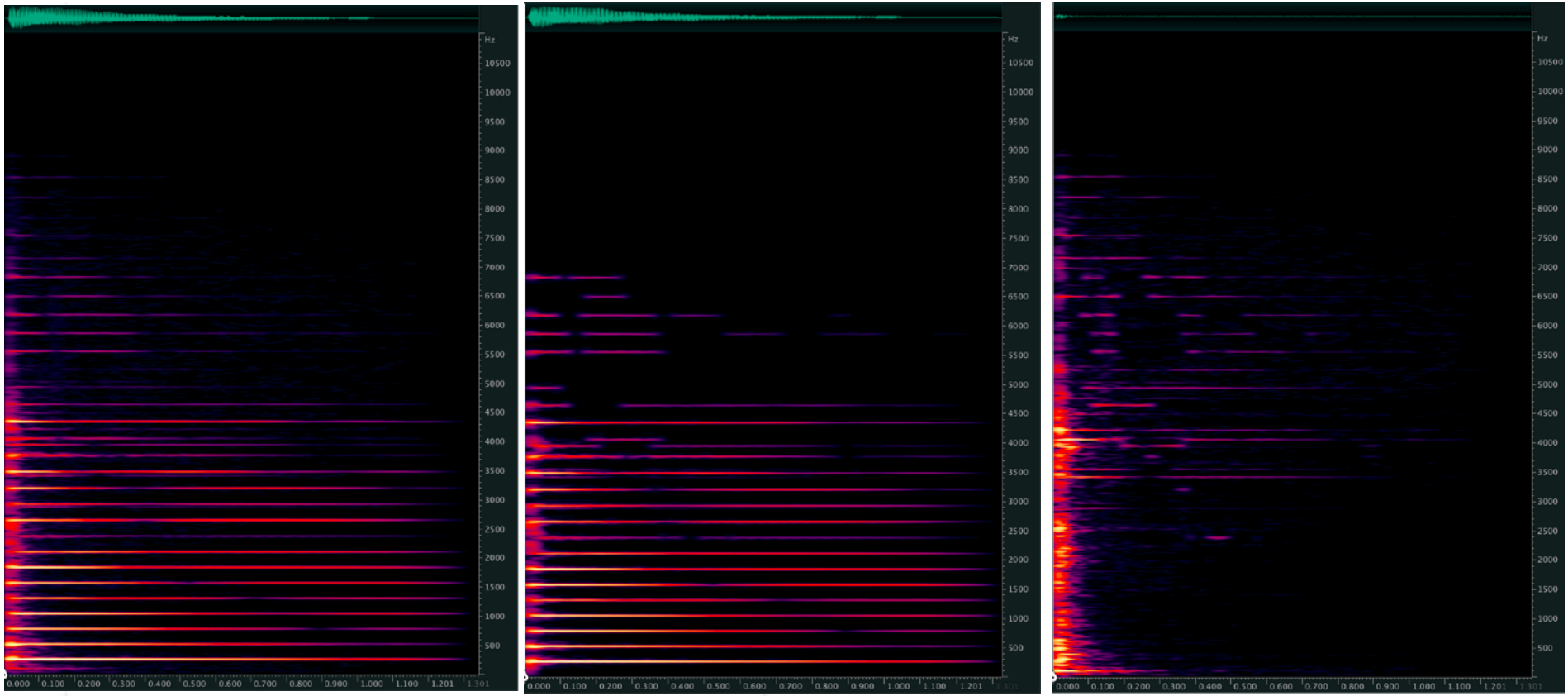
- $x(t)$ is a single source, mono-phonic, harmonic sound
- exemple: voiced speech, musical instruments
- $f_h(t_m) = hf_0(t_m)$
  - If we know $f_0(t_m)$ (pitch tracking), we can deduce the frequencies of the sinusoidal components: $f_h(t_m) = hf_0(t_m)$,
  - We only need to estimate the amplitude and initial phases: $A_h(t_m)$ et le $\phi_h(t_m)$
  - $H(t)$ is constant over time
  - No needs for partial tracking



Sinusoidal (harmonic) model

Examples: piano

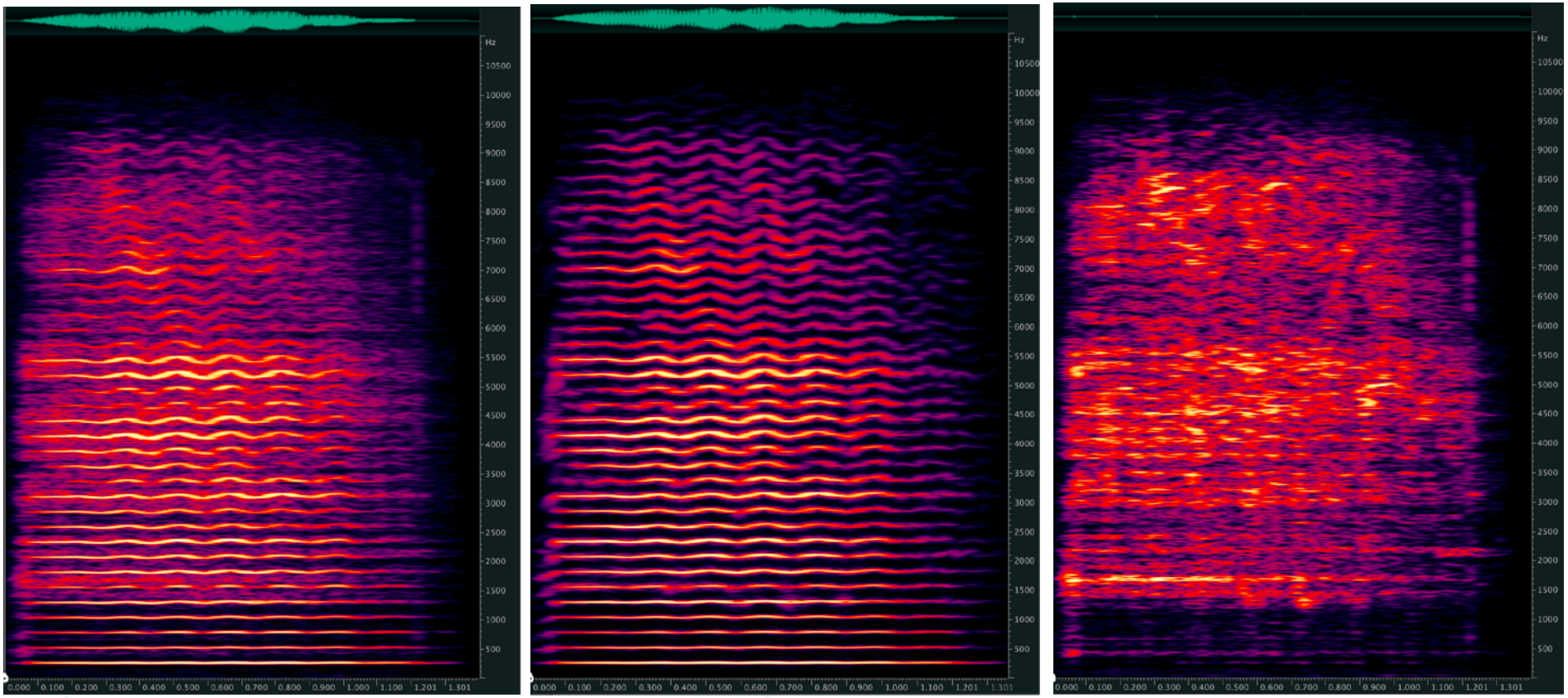
$x(t)$ $\hat{x}(t)$ $b(t) = x(t) - \hat{x}(t)$



Sinusoidal (harmonic) model

Examples: violin

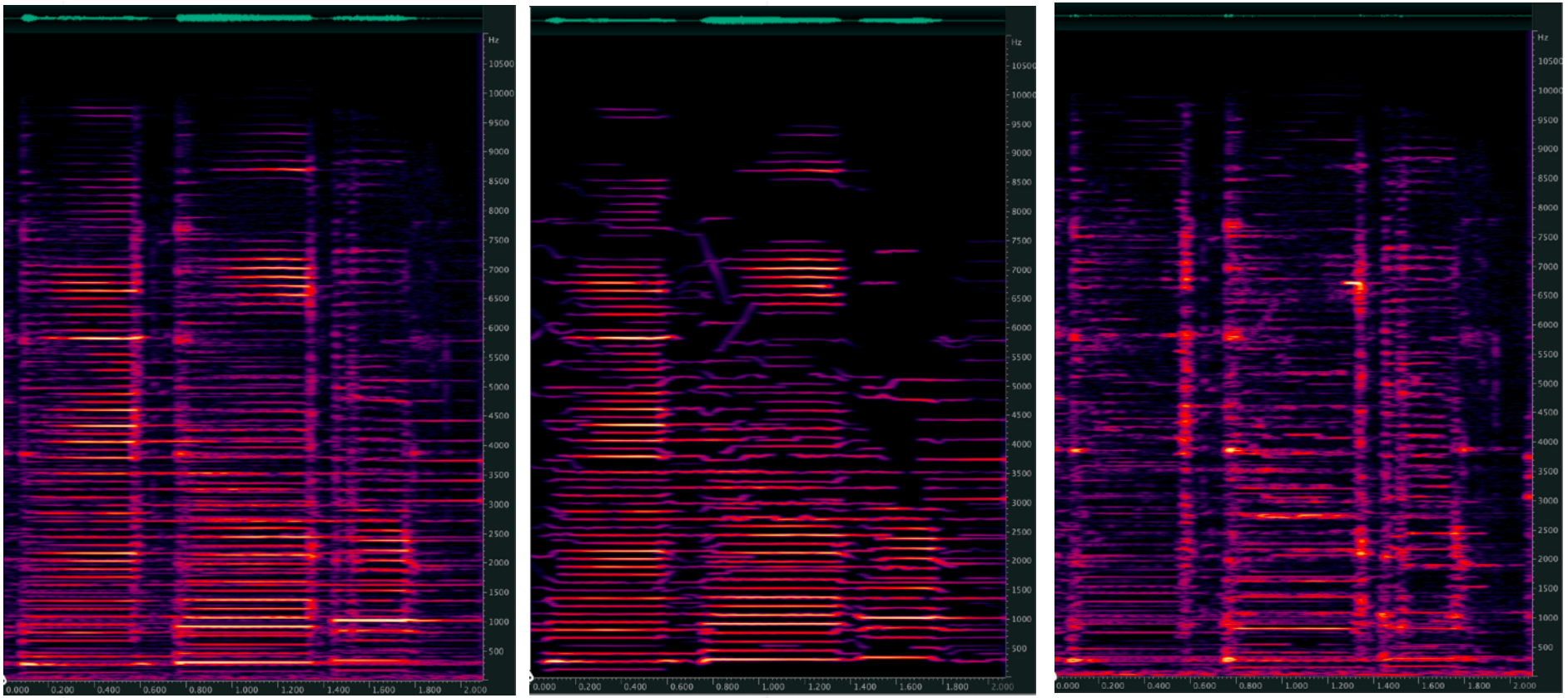
$x(t)$ $\hat{x}(t)$ $b(t) = x(t) - \hat{x}(t)$



Sinusoidal (harmonic) model

Examples: sitar

$x(t)$ $\hat{x}(t)$ $b(t) = x(t) - \hat{x}(t)$



Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

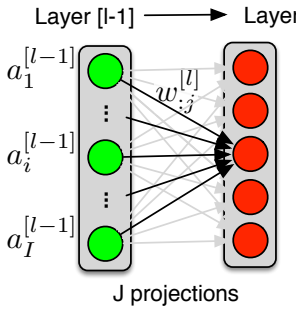
Application: multi-f0 estimation, DDSP

Multi-Layer-Perceptron (MLP)

- Extension of the Perception

- Neurons are organized into Layers which are Fully-Connected
  - $a_j^{[l]}$ is connected to all neurons $a_i^{[l-1]}$
  - connection done through
    - multiplications by weights $w_{ij}^{[l]}$,
    - addition of a bias $b_j^{[l]}$,
    - passing through non-linear activation $g(.)$
  - Each $\mathbf{w}_j^{[l]}$ defines a specific projection of the previous layer

Fully Connected Neural Network



$$a_j^{[l]} = g(\sum_i a_i^{[l-1]} w_{ij}^{[l]} + b_j^{[l]})$$

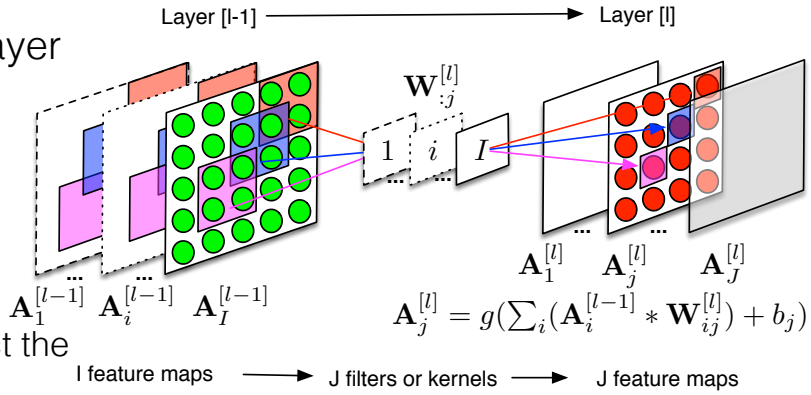
$$\mathbf{a}^{[l]} = g(\mathbf{a}^{[l-1]} \mathbf{W}^{[l]} + \mathbf{b}^{[l]})$$

[Frank Rosenblatt, "**The perceptron, a perceiving and recognizing automaton**" Project Para. Cornell Aeronautical Laboratory, 1957] LINK

Convolutional Neural Network (CNN)

- (1) Assume local connectivity of neurons of a given layer (vision: nearby pixels more correlated)
  - Each region (x, y) of $\mathbf{A}_i^{[l-1]}$ is projected individually
  - Results= feature map noted $\mathbf{A}_{i \to j}^{[l]}$ for the j^{th} projection
- (2) Add a **parameter sharing** property
  - for a given j, the same projection $\mathbf{W}_{ij}^{[l]}$ is used to project the different regions $(x, y) \rightarrow$ the weights are shared
  - apply the same projection to the various regions (x, y)
- (1)+(2) $\rightarrow$ convolution operator
- In practice
  - several input feature maps $\mathbf{A}_{1 \ldots i \ldots I}^{[l-1]}$, projection with a tensor $\mathbf{W}_{:j}^{[l]}$ (extends over I)
  - Results: feature map $\mathbf{A}_j^{[l]}$
  - Several projections: J different convolutions resulting in J output feature maps
- To reduce dimensionality, allows spatial invariance:
  - **max-pooling**

Convolutional Neural Network



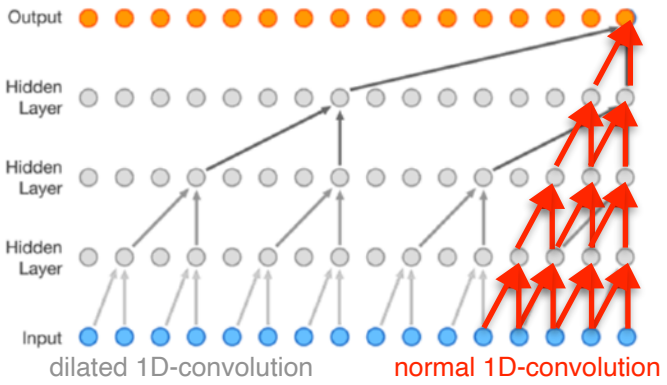
[K. Fukushima et al., "**Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition**". In Competition and cooperation in neural nets, 1982]
[Y. LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haner. "**Gradient- based learning applied to document recognition**". Proceedings of the IEEE, 86(11):2278–2324, 1998]

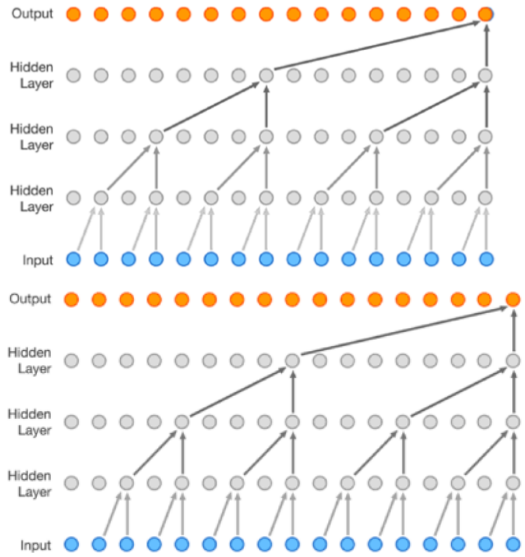
Temporal Convolutional Networks (TCN)

- Motivation: learn better projections than Fourier
- 1D-convolution applied on the raw audio waveform $x(n)$
  - filters $\mathbf{W}_j$ have only one dimension (time)
  - convolution is done over time

- Receptive field (RC)
  - portion of the input data to which a given neuron responds
  - images (256x256): only a few layers is necessary in CV to make the RC of a neuron cover the whole input image
  - audio (44100/sec): requires a huge number of layers

- 1D-Dilated- Convolutions
  - $(x \circledast_d w)(n) = \sum_{i=0}^{l-1} w(i)x(n-(d\,.\,i))$
  - the filters is convolved with the signal only considering one over d values





[A. van den Oord et al. "**Wavenet: A generative model for raw audio**", 2016] LINK

Temporal Convolutional Networks (TCN)

- 1D-Convolution
- + causality constraint
- + stacks two dilated-convolutions on top of each other (with weight-normalization, ReLu, DropOut)
- + with a parallel residual path

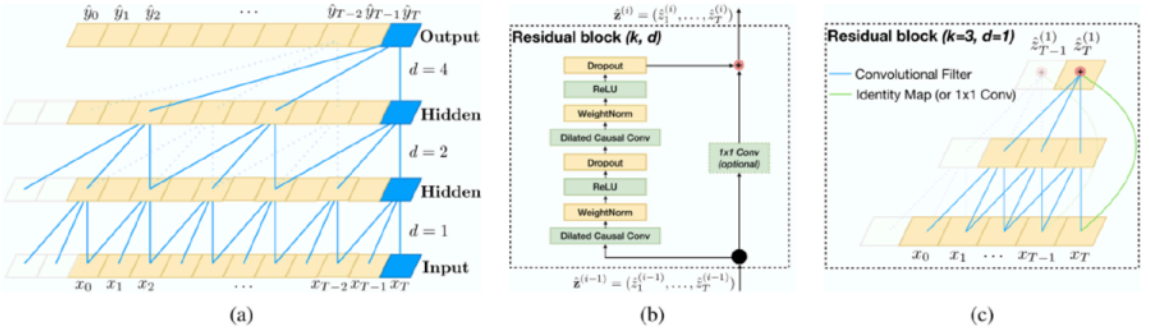


Figure 1. Architectural elements in a TCN. (a) A dilated causal convolution with dilation factors $d = 1, 2, 4$ and filter size $k = 3$. The receptive field is able to cover all values from the input sequence. (b) TCN residual block. An 1x1 convolution is added when residual input and output have different dimensions. (c) An example of residual connection in a TCN. The blue lines are filters in the residual function, and the green lines are identity mappings.

[Shaojie Bai et al. "**An empirical evaluation of generic convolutional and recurrent networks for sequence modeling**", 2018] LINK

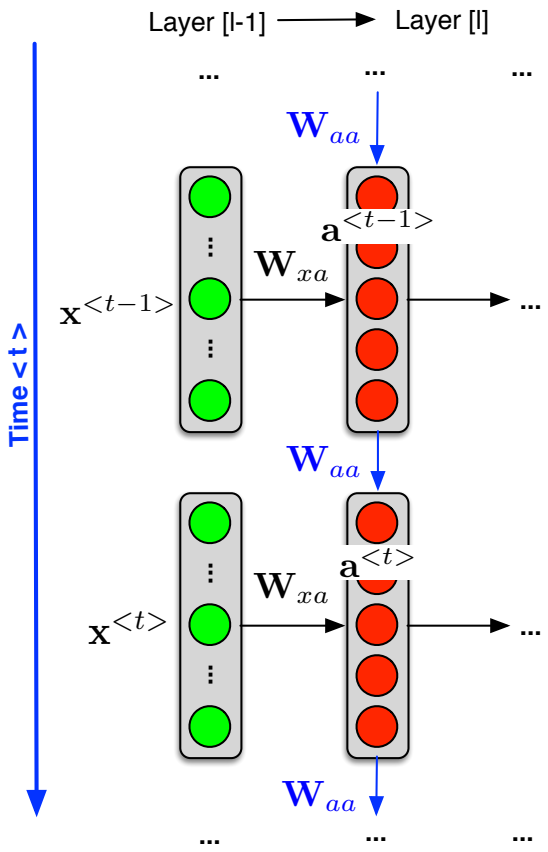
Recurrent Neural Network (RNN)

- RNNs take into account the sequential aspect of the data
  - RNNs have a memory that keeps track of previously proposed events of the sequence
  - Internal/hidden representation of the data at time t, $\mathbf{a}^{<t>}$ does not only depend on the input data $\mathbf{x}^{<t>}$ but also on the internal/ hidden representation at the previous time $\mathbf{a}^{<t-1>}$

- More sophisticated RNN cells
  - Long Short Term Memory (LSTM)
  - Gated Recurrent Units (GRU)

Recurrent Neural Network

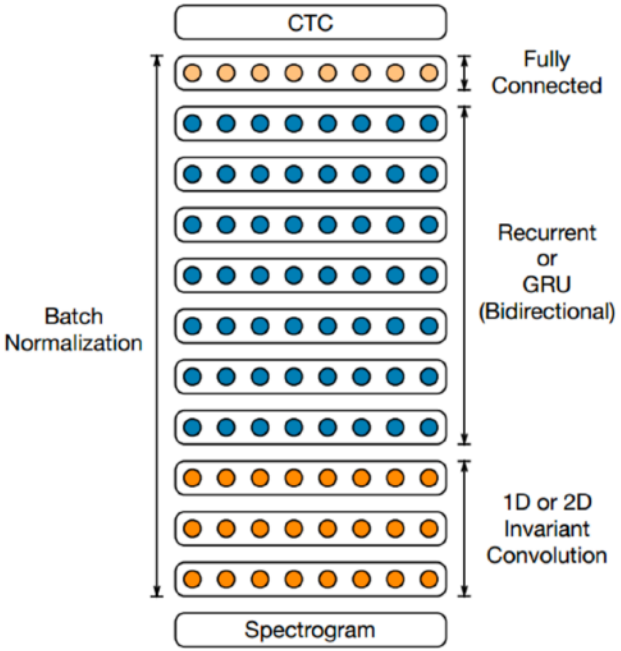
$$\mathbf{a}^{<t>} = g(\mathbf{x}^{<t>}\mathbf{W}_{xa} + \mathbf{a}^{<t-1>}\mathbf{W}_{aa} + \mathbf{b}_a)$$



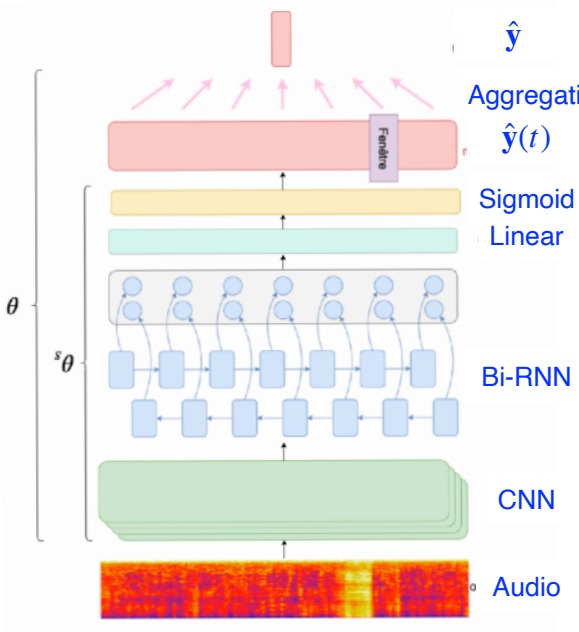
[David E Rumelhart et al. "**Learning representations by back-propagating errors**". Nature, 323(6088):533–536, 1986]
[Sepp Hochreiter and Jurgen Schmidhuber. "**Long short-term memory**". Neural computation, 9(8):1735–1780, 1997]
[Kyunghyun Cho et al. "**Learning phrase representations using RNN encoder-decoder for statistical machine translation**", 2014]

Typical audio architecture: RCNN = CNN followed by a RNN

- CNN (Conv1D or Conv2D) used for
  - feature extraction
- RNN used for
  - temporal smoothing
  - language model



ASR Baidu Speech 2



DCASE Task 4, Turpault baseline

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

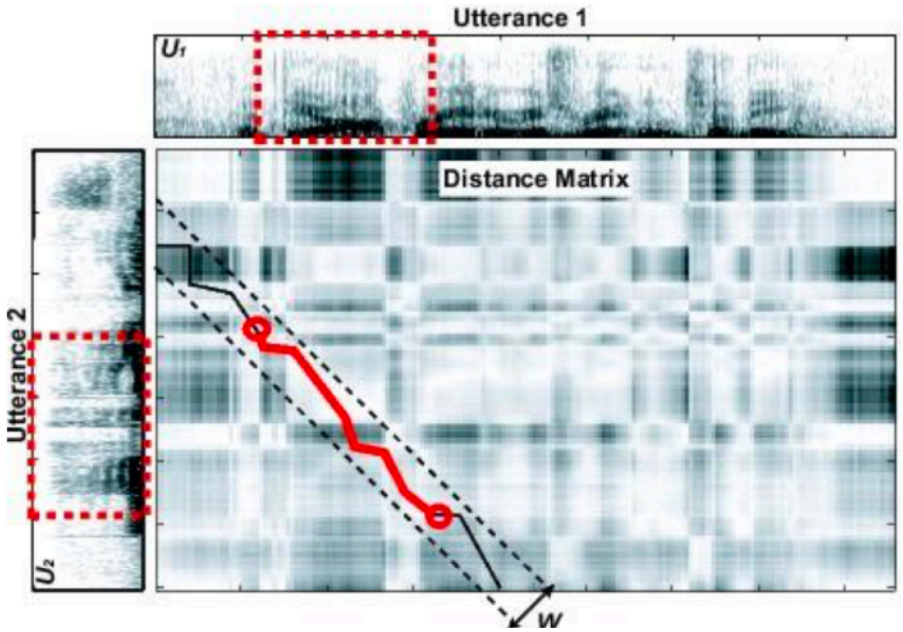
Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

History

- **Early 1970s**
  - Dynamic Time Warping (DTW) to handle time
  - Distance measure for spectral variability

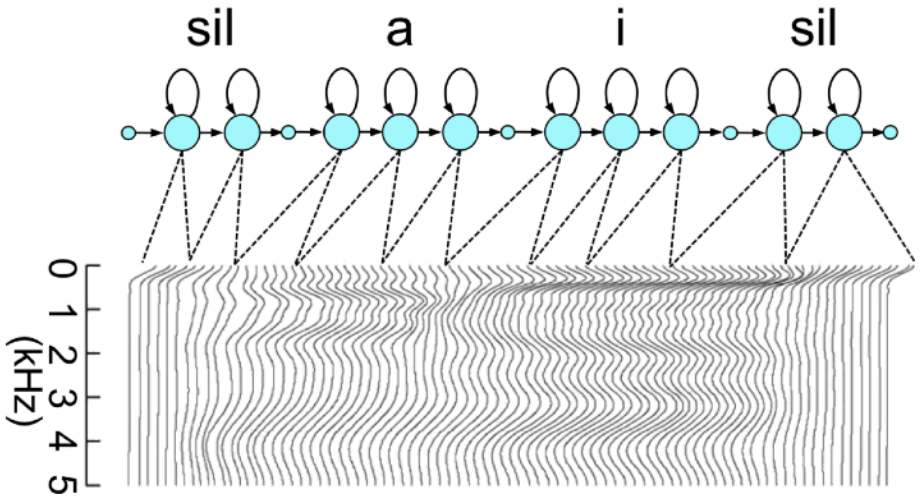


source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

ASR: Automatic Speech Recognition

History

- **Mid-Late 1970s**
  - Hidden Markov Models (HMMs)
    - Statistical models of spectral variatior
- **Mid 1980s**
  - HMMs become the dominant technique for

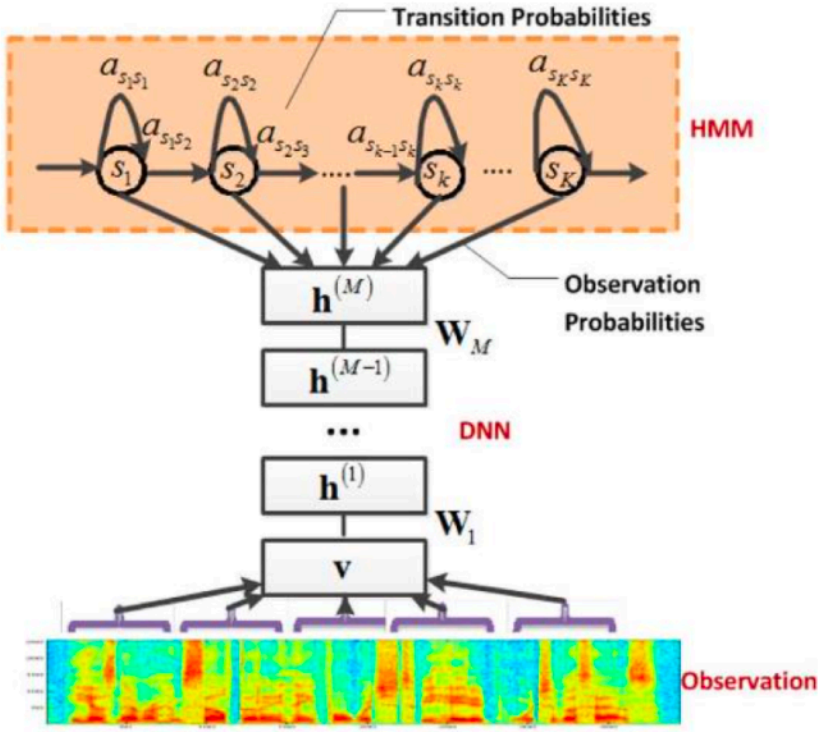


source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

ASR: Automatic Speech Recognition

History

- **1990s**
  - Large vocabulary continuous dictation
- **2000s**
  - Discriminative training (minimize word/phone error r
- **2010s**
  - Deep learning significantly reduce error rate



source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

Speech: Automatic Speech Recognition

Traditional approach: GMM-HMM

Traditional approach: GMM-HMM

- **Goal of Automatic Speech Recognition ?**
  - Find the most likely sentence (word sequence) W, which transcribes the speech audio A

$$\hat{W} = \arg\max_{W} P(W \mid A)$$
$$= \arg\max_{W} P(A \mid W) P(W)$$

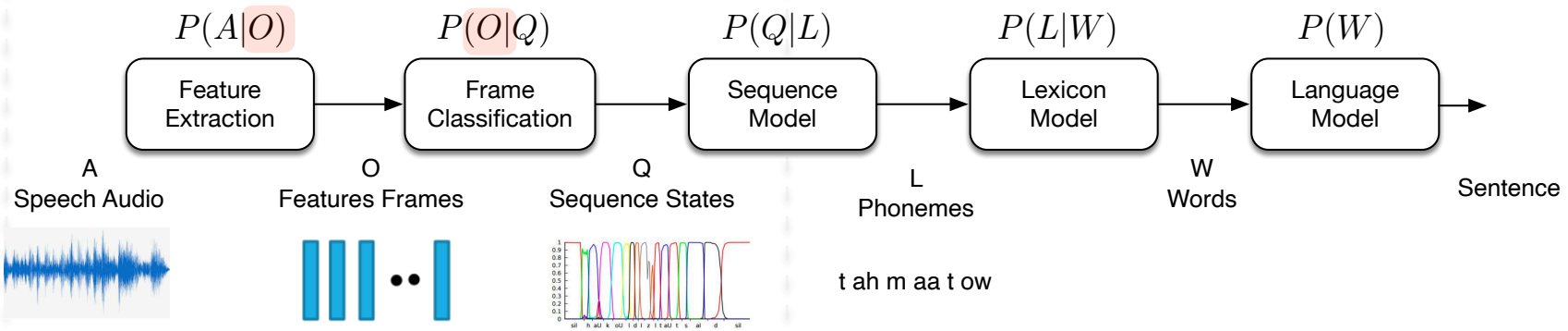
    - $P(A \mid W)$: the **acoustic** model
    - $P(W)$: the **language** model

- **Training ?**
  - Find the parameters of the acoustic and language models separately
    - Speech Corpus : speech waveform and human-annotated transcriptions
    - Language model : with extra data (prefer daily expressions corpus for spontaneous speech)

Traditional approach: GMM-HMM

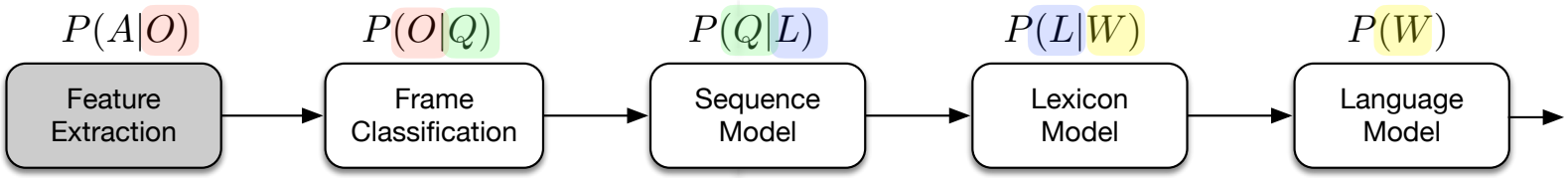
– **Architecture of Speech Recognition**

$$\hat{W} = \arg\max_W P(W|A)$$
$$= \arg\max_W P(A|W)P(W)$$
$$= \arg\max_W P(A|O)P(O|Q)P(Q|L)P(L|W)P(W)$$

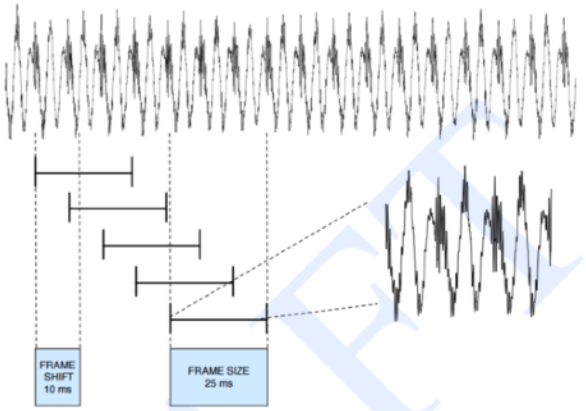


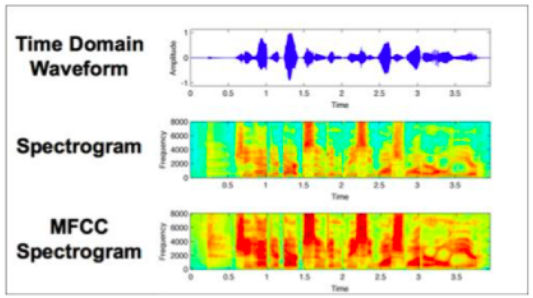
source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

Traditional approach: GMM-HMM



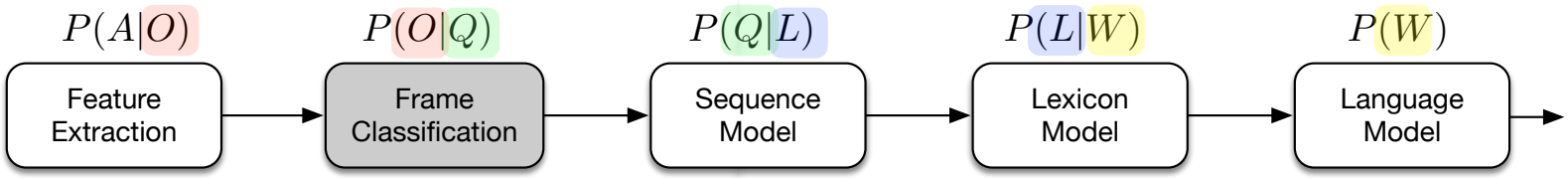
- $P(A \mid O) = P(Audio \mid Features)$
- **Feature extraction**
  - Raw waveforms are transformed into a sequence of featu approaches
  - Time domain to frequency domain





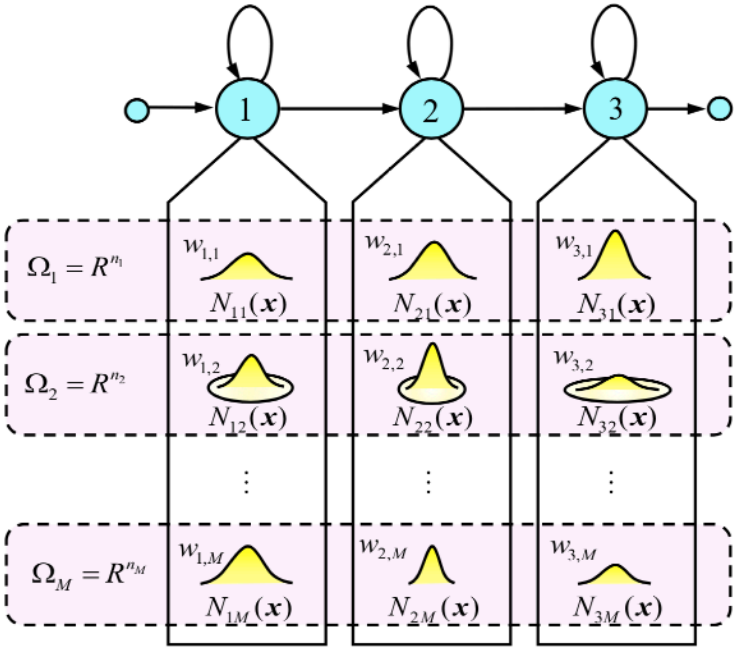
source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

Traditional approach: GMM-HMM



- $P(O|Q) = P(Features|States)$
- **Frame Classification**
  - Gaussian Mixture Model (GMM)
    - Output probability is modeled by a mixture c

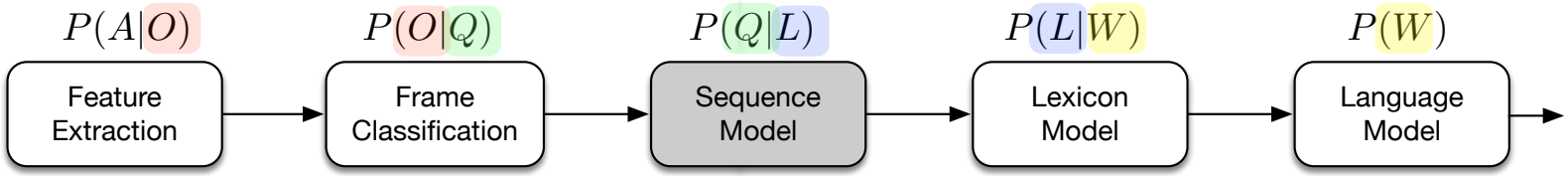
$$b_q(o_t) = b(o_t|q_t)$$
$$= \sum_{m=1}^{M} \mathcal{N}(o_t|\mu_m, \Sigma_m)$$



source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

ASR: Automatic Speech Recognition

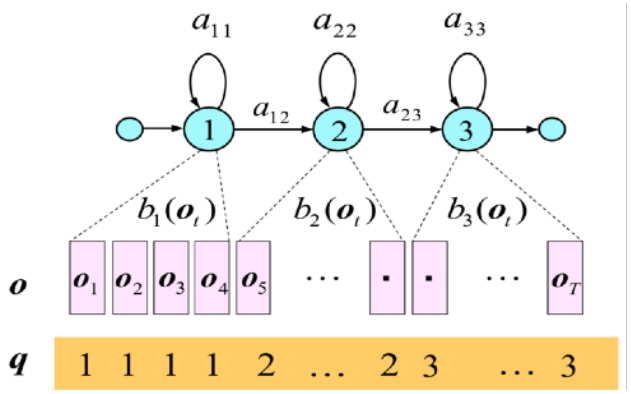
Traditional approach: GMM-HMM



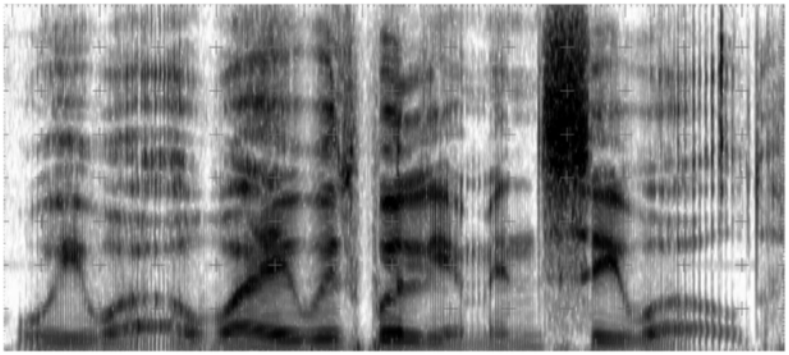
- $P(Q|L) = P(States|Phonemes)$
- **Sequence model**
  - Hidden Markov Models (HMM)
    - The Markov chain whose state sequence is u

      a_{ij}: state transition probability

      $b_q(o_t)$: output probability



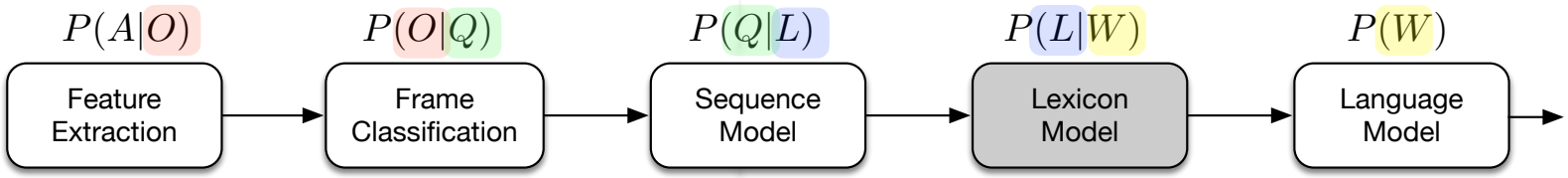
- Context Dependent Model
  - "We **w**ere away with William in Sea **W**orld"



  - Realization of **w** varies but similar patterns occur in the similar context

source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

Traditional approach: GMM-HMM



- $P(L|W) = P(Phonemes|Words)$
- **Lexicon model**
  - Lexical modelling forms the bridge between the acoustic and language models
  - Prior knowledge of language
  - Mapping between words and the acoustic units (phoneme is most common)

Deterministic

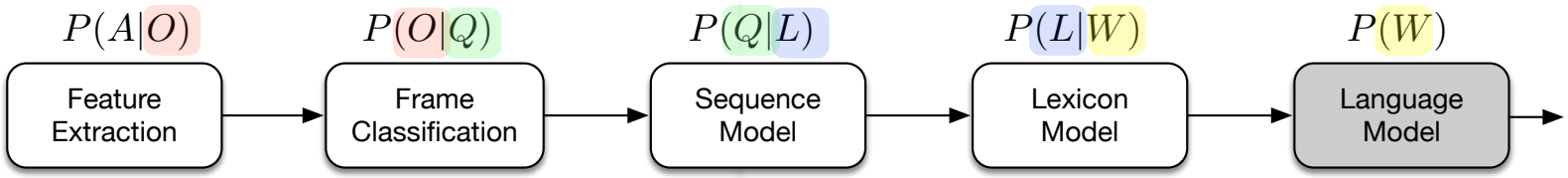
Word	Pronunciation
TOMATO	t ah m aa t ow
	t ah m ey t ow
COVERAGE	k ah v er ah jh
	k ah v r ah jh

Probabilistic

Word	Pronunciation	Probability
TOMATO	t ah m aa t ow	0.45
	t ah m ey t ow	0.55
COVERAGE	k ah v er ah jh	0.65
	k ah v r ah jh	0.35

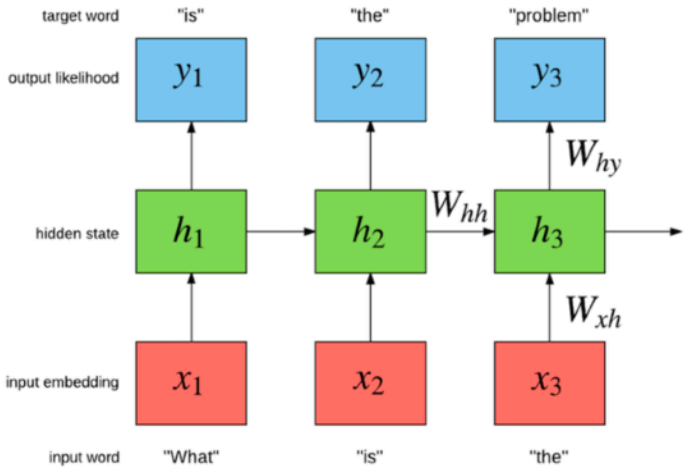
source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

Traditional approach: GMM-HMM



- $P(W) = P(Words)$
- **Language model**
  - Language model is a probabilistic model used to
    - Guide the search algorithm (predict next word given history)
    - Disambiguate between phrases which are acoustically similar
      - "Great wine" vs "Grey twine"
  - It assigns probability to a sequence of tokens to be finally recognized
  - N-gram model $P(W_N | w_1, w_2, \ldots, w_{N-1})$

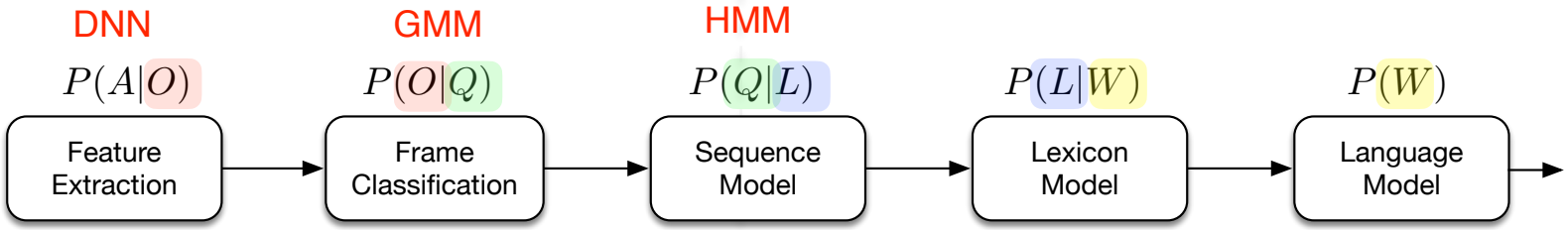
- using bi-grams, tri-grams
- using a Recurrent neural network



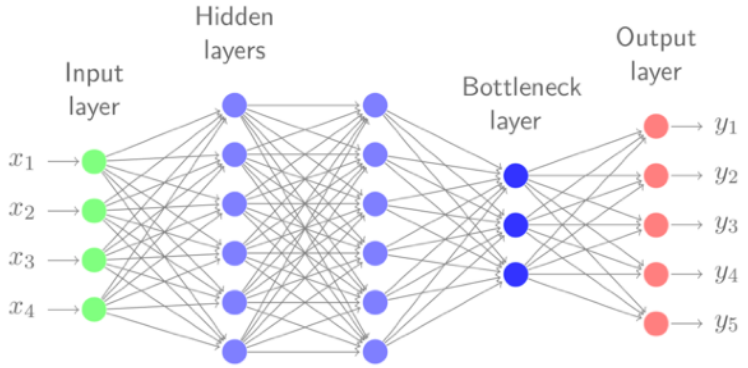
source : Yuchen Fan, Matt Potok, Christopher Shroba, "CS 598 LAZ : Cutting-Edge Trends in Deep Learning and Recognition"

Speech: Automatic Speech Recognition
Deep-learning approaches

Deep learning approaches: 1) DNN ⇒ GMM ⇒ HMM



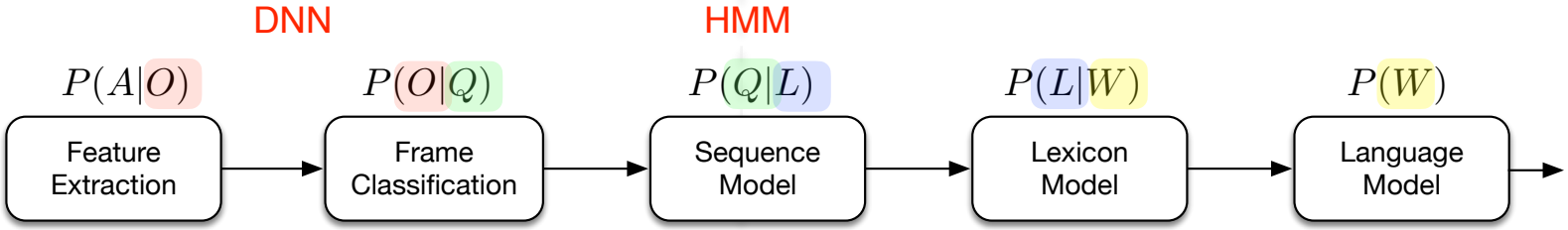
- **DNN / GMM / HMM**
  - Use neural networks to compute nonlinear feature rep
    - Trained audio features
    - Tandem, Bottleneck, DNN-derived features
    - Low-dimensional representation is modeled conventionally with GMMs
    - Allows all the GMM machinery and tricks to be exploited



- **How?**
  - Train a neural network to discriminate classes.
  - Use output or a low-dimensional bottleneck layer representation as features.
- **TRAP :**
  - Concatenate PLP-HLDA features and NN features.

H. Hermansky et al. "Tandem connectionist feature extraction for conventional HMM systems", 2000, IEEE ICASSP

Deep learning approaches: 1) DNN ⇒ HMM



– **DNN / HMM**

- Use neural networks (DNN) instead of GMM to estimate phonetic unit probabilities
  - Perform frame classification using a DNN
  - Classify acoustic features for state labels
  - Take softmax output as a posterior $P(state \mid o_t) = p(q_t \mid o_t)$
  - Work as output probability in HMM

$$p(o_t \mid q_t) = \frac{p(q_t \mid o_t) p(o_t)}{p(q_t)}$$

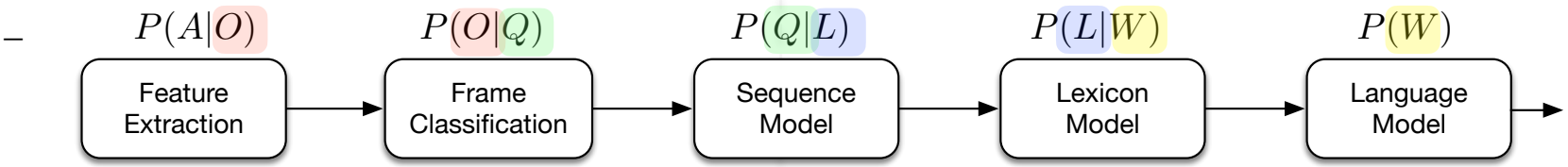
- where $p(q_t)$ is the prior probability for states

- Train the network as a classifier with a softmax across the phonetic units.
  - Train with cross-entropy.
  - Softmax $y(i) = \frac{\exp(a_\theta(i))}{\sum_{j=1}^{N} \exp(a_\theta(j))}$
  - will converge to posterior across phonetic states $p(q_i \mid o_t)$

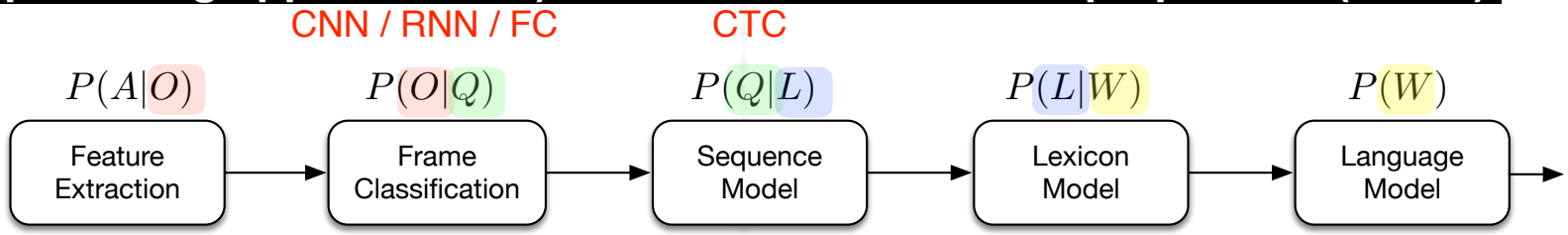
G. Hinton, et al. "Deep Neural Networks for Acoustic Modeling in Speech Recognition". 2012, Signal Processing Magazine

Deep learning approaches: 3) End-to-End models: LSTM CTC

CNN / RNN / FC CTC

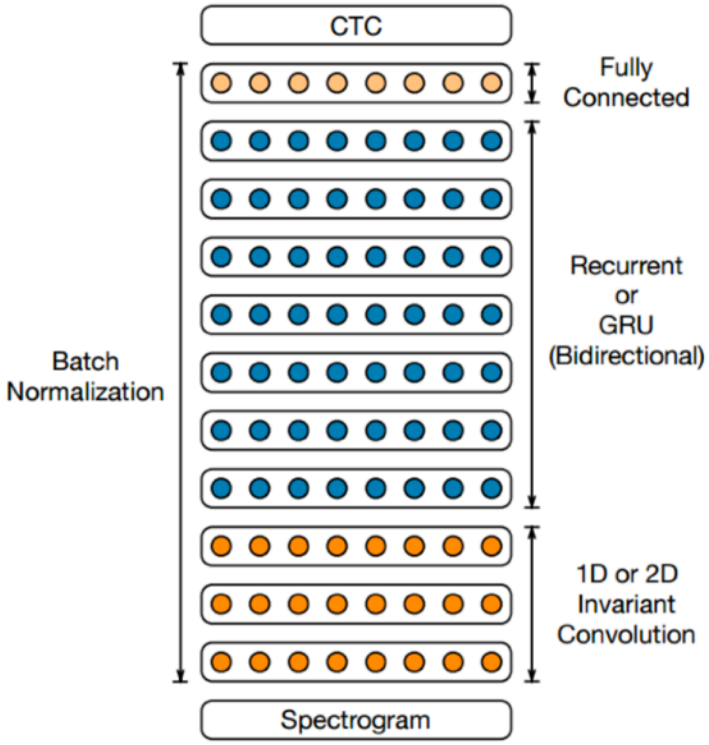


Deep learning approaches: 3) End-to-End models: Deep Speech 2 (Baidu)



- 3 layers of 2D-invariant convolution
- 7 layers of bidirectional simple recurrence 100M para
- Word error rates

Test set	Deep speech 2	Human
WSJ eval'92	3.60	5.03
WSJ eval'93	4.98	8.08
LibriSpeech test-clean	5.33	5.83
LibriSpeech test-other	13.25	12.69

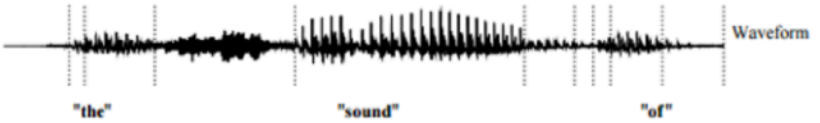


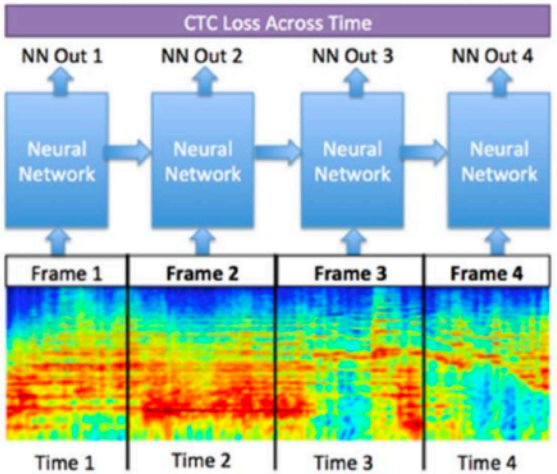
Connectionist Temporal Classification

Connectionist Temporal Classification

Goal

- end-to-end training
  - from audio/spectrogram to characters
- can be combined with a language model
- neural network is usually a stack of bi-LSTM





Problem

- for training and decoding :
  - output Y (the characters) need to be aligned with input X(the audio/spectrogram)
- in practice input and output have different length ($T_y \neq T_x$)
- manual alignment very costly (cannot at scale)

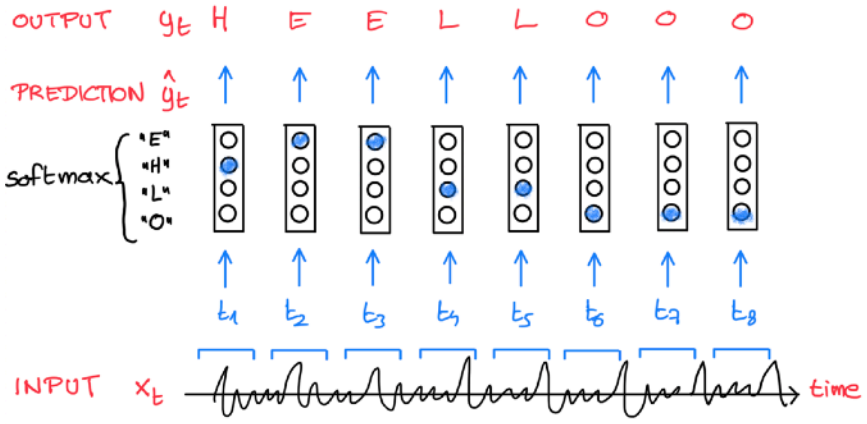
Solution

- Connectionist temporal classification
- Transform the output such that $T_y = T_x$

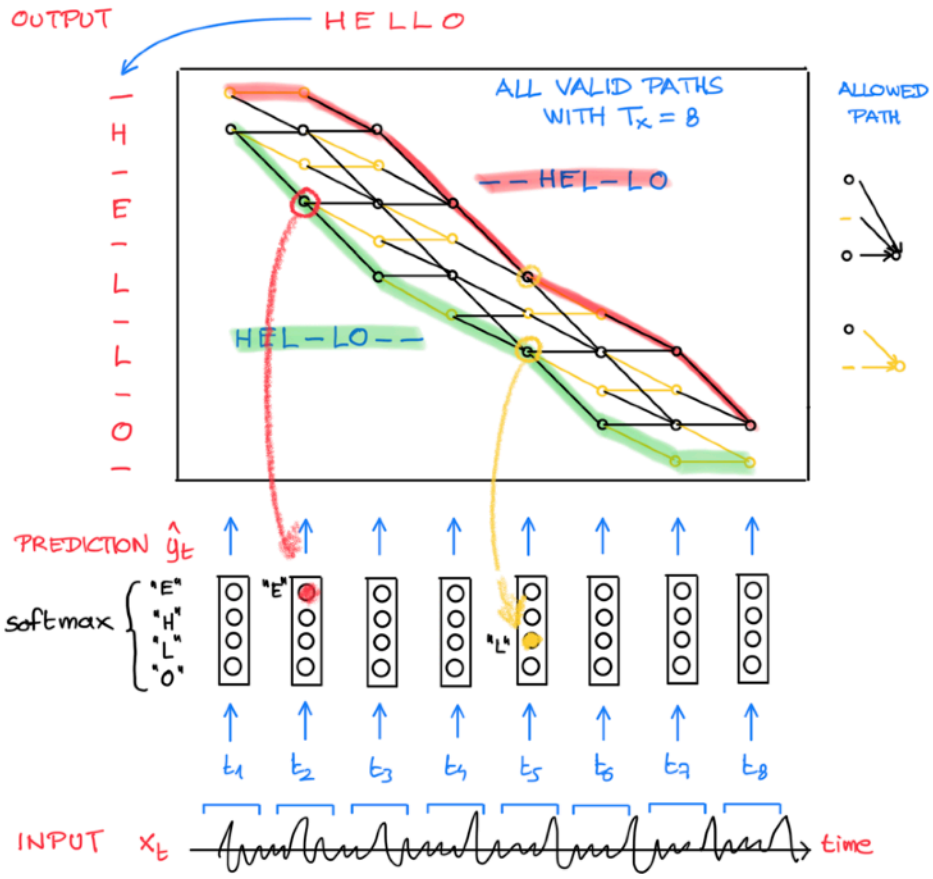
[A. Graves et al."Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks", ICML, 2006] Link
[W. Gharbieh, "Connectionist Temporal Classification, Labelling Unsegmented Sequence Data with RNN"] Link
[A. Hannun "Sequence Modeling With CTC"] https://distill.pub/2017/ctc/

Connectionist Temporal Classification

Strongly aligned



Weakly aligned data

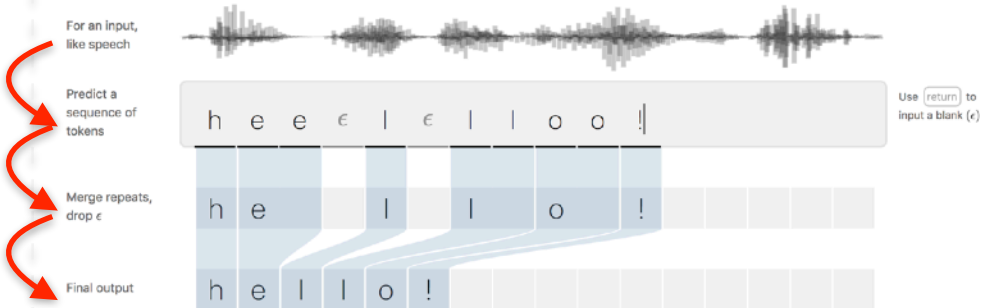


[A. Graves et al."Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks", ICML, 2006] Link

Decoding

- **Network**:
  - $\mathbf{y} = f_\theta(\mathbf{x}) \quad (\mathbb{R}^m)^T \mapsto (\mathbb{R}^n)^T$
  - y_t^k probability of observing output unit k at time t
- y_t^k over the alphabet $L' = L \cup \{blank\}$
  - L={a, b, ..., z}: the possible characters
  - + a blank symbol ("ϵ" or "-")
- To get the final text
  - define a many-to-one **mapping function** $\mathcal{B}$: transforms the sequence by merging identical characters if not separated by "ϵ" or "-"

- **Examples** :
  - $\mathcal{B}$("hell-loo") = $\mathcal{B}$("hel-lo") = "hello"
  - $\mathcal{B}$("hellllloo") = "helo"
  - $\mathcal{B}$("cc-a-tt") = $\mathcal{B}$("c-a-t") = "cat"
  - $\mathcal{B}$("-c-a-t–") = $\mathcal{B}$("-c-a-t-") = "cat"
  - $\mathcal{B}$("c-aaa-at") = $\mathcal{B}$("c-a-at") = "caat"

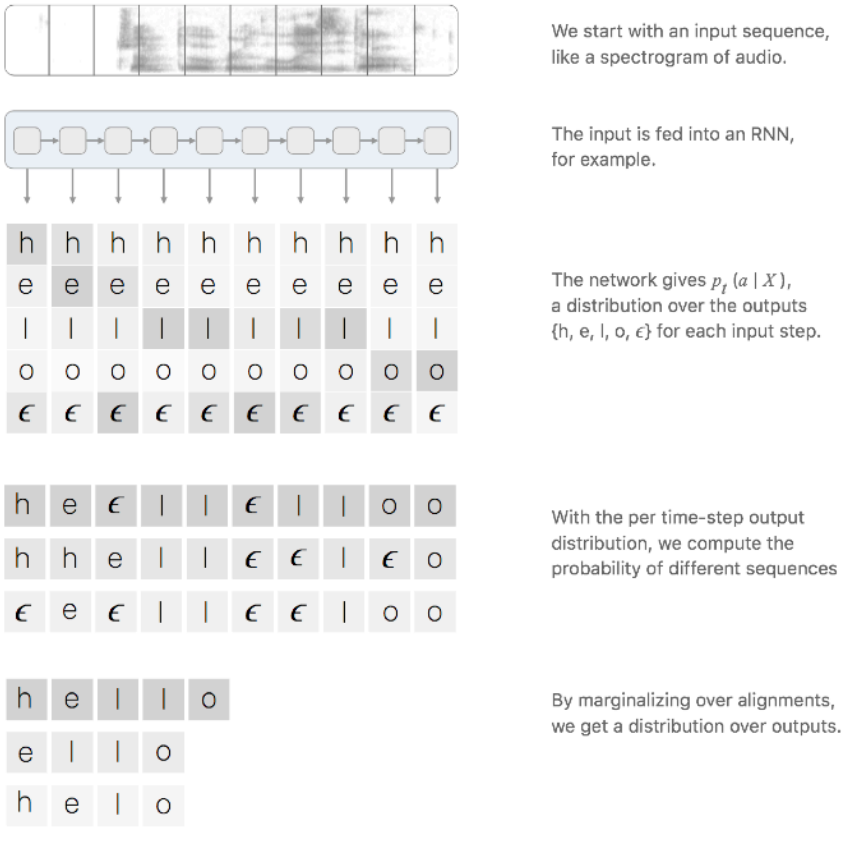


Training: maximise the probability $p(\mathbf{l}\,|\,\mathbf{x}) \Rightarrow$ minimise the CTC loss

- $p(\mathbf{l}\,|\,\mathbf{x})$?
  - we want to know how likely a given sentence $\mathbf{l}$ ="hello" is, given the input $\mathbf{x}$=spectrogram
  - T_x and T_y have different length
- **Alignements paths π**
  - we need to define the different possible "pronunciations" of $\mathbf{l}$="hello" such that it matches T_x
  - $\mathcal{B}^{-1}$("hello") for $T_x = T_y = 8$
    - "hel-looo", "hel-lloo", "hhhel-lo", "hheel-lo", . . .
- $p(\mathbf{l}\,|\,\mathbf{x})$
  - We sum up over all possible paths $\pi \in \mathcal{B}^{-1}(\mathbf{l})$

$$p(\mathbf{l}\,|\,\mathbf{x}) = \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{l})} p(\pi\,|\,\mathbf{x})$$

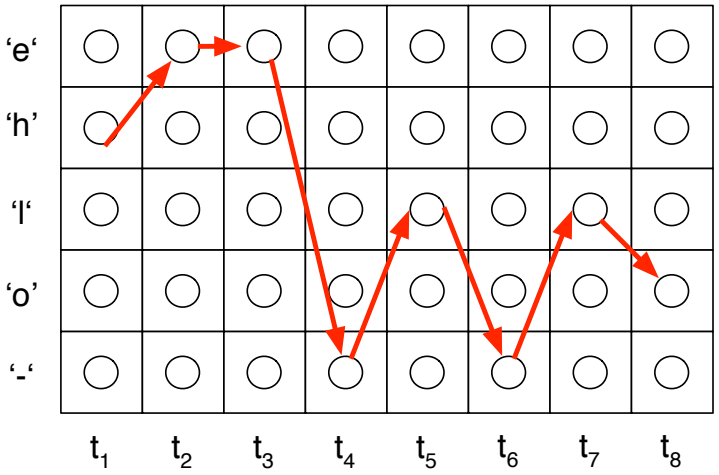
$$p(\pi\,|\,\mathbf{x}) = \Pi_{t=1}^{T} y_t^{\pi_t}$$



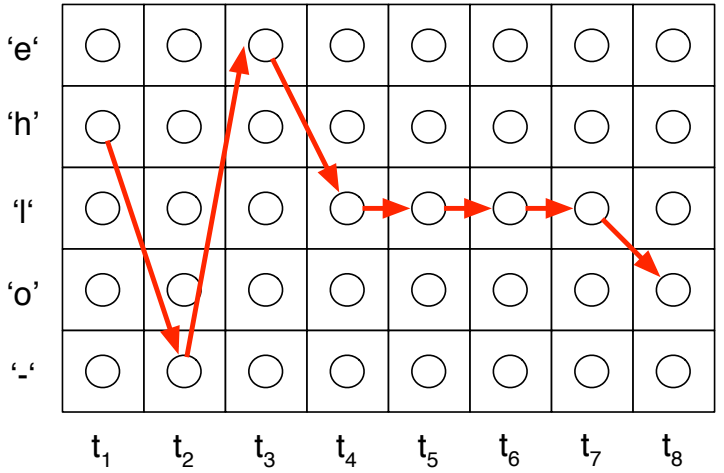
Training : minimise the CTC Loss

- CTCLoss = $-\log p(\mathbf{l})$
- **Example:**
  - for the world "hello", we wand to minimise $-\log p(''hello'')$
- **Problem:**
  - the number of alignement paths $\pi \in \mathcal{B}^{-1}(\mathbf{l})$ can quickly explode !
  - there are $5^8 = 390.625$ ways (possible paths) to go from t_1 to t_8
  - cannot be solved trivially

f('hee-l-lo')='hello'

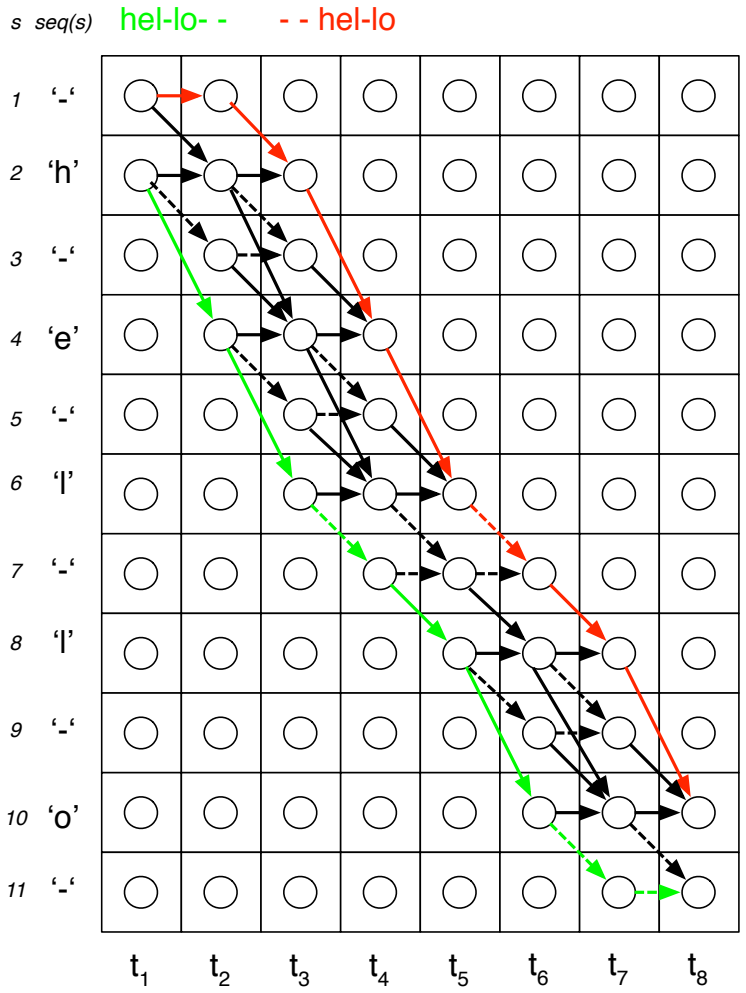


f('h-elllllo')='helo'

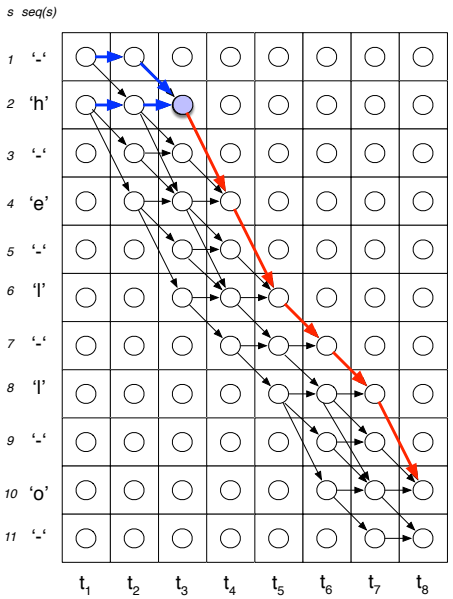


Training : the CTC Loss

- All possible paths $\pi \in \mathcal{B}^{-1}(\mathbf{l})$
  - Capture variations in pronunciation from $\mathcal{B}$("hel-lo--") to f("--hel-lo")



Training : the CTC Loss



– Complete path calculation example

– The **forward pass** gives

$$\alpha_{t_3}(2) = p(''--h'') + p(''-hh'') + p(''hhh'')$$

- $$= y_{t_1}^{''-''} y_{t_2}^{''-''} y_{t_3}^{''h''} + y_{t_1}^{''-''} y_{t_2}^{''h''} y_{t_3}^{''h''} + y_{t_1}^{''h''} y_{t_2}^{''h''} y_{t_3}^{''h''}$$

– The **backward pass** results in

$$\beta_{t_3}(2) = p(''hel-lo'')$$

- $$= y_{t_3}^{''h''} y_{t_4}^{''e''} y_{t_5}^{''l''} y_{t_6}^{''-''} y_{t_7}^{''l''} y_{t_8}^{''o''}$$

– The **final results**

$$\alpha_{t_3}(2)\beta_{t_3}(2) = y_{t_1}^{''-''} y_{t_2}^{''-''} y_{t_3}^{''h''} \cdot y_{t_3}^{''h''} y_{t_4}^{''e''} y_{t_5}^{''l''} y_{t_6}^{''-''} y_{t_7}^{''l''} y_{t_8}^{''o''}$$
$$+ y_{t_1}^{''-''} y_{t_2}^{''h''} y_{t_3}^{''h''} \cdot y_{t_3}^{''h''} y_{t_4}^{''e''} y_{t_5}^{''l''} y_{t_6}^{''-''} y_{t_7}^{''l''} y_{t_8}^{''o''}$$
$$+ y_{t_1}^{''h''} y_{t_2}^{''h''} y_{t_3}^{''h''} \cdot y_{t_3}^{''h''} y_{t_4}^{''e''} y_{t_5}^{''l''} y_{t_6}^{''-''} y_{t_7}^{''l''} y_{t_8}^{''o''}$$

- $$= \left[p(''--hel-lo'') + p(''-hhel-lo'') + p(''hhhel-lo'')\right] \cdot y_{t_3}^{''h''}$$

– Total probability of all paths going through $''h''$ at t_3

- $$\frac{\alpha_{t_3}(2)\beta_{t_3}(2)}{y_{t_3}^{''h''}}$$

Training : the CTC Loss

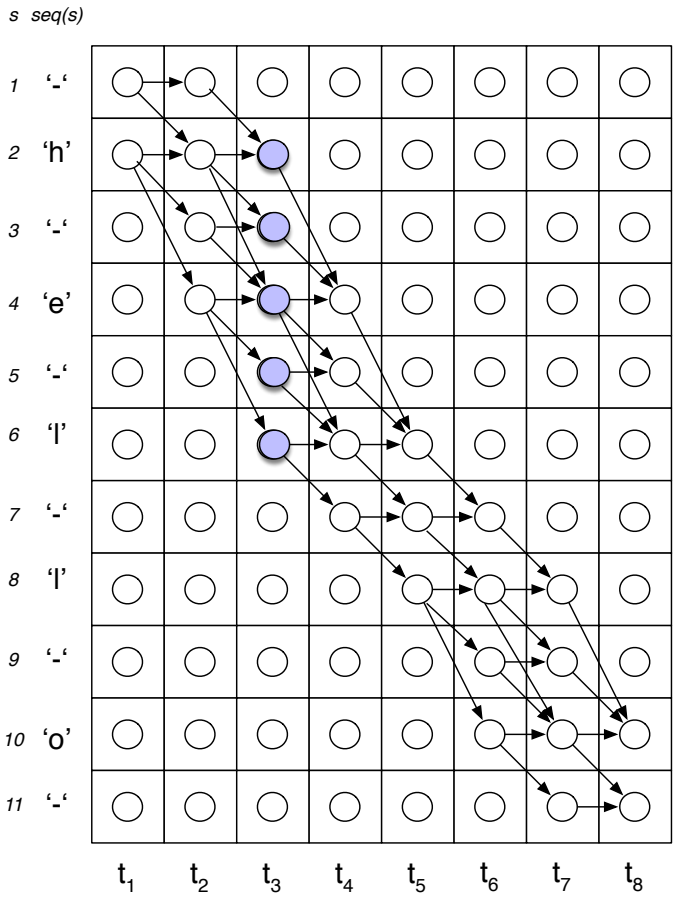
– **Calculating the Loss at a particular time-step example**

– $p(''hello'')$ at $t = 3$ is the sum of the probabilities of all paths through all the symbols

- $$p(''hello'') = \sum_{s=2}^{6} \frac{\alpha_{t_3}(s)\beta_{t_3}(s)}{y_{t_3}^{seq(s)}}$$

– In general, the probability of the ground label is

- $$p(''hello'') = \sum_{s=1}^{|seq|} \frac{\alpha_t(s)\beta_t(s)}{y_t^{seq(s)}}$$



Training : the CTC Loss

– **How to do back-propagation ?**

- $$\frac{\delta - \log p(''hello'')}{\delta y_t^k} = \frac{-1}{p(''hello'')} \frac{\partial p(''hello'')}{\partial y_t^k}$$

– since

- $$p(''hello'') = \sum_{s=1}^{|seq]} \frac{\alpha_t(s)\beta_t(s)}{y_t^{seq(s))}}$$

– we have

- $$\frac{\partial p(''hello'')}{\partial y_t^k} = \frac{1}{(y_t^k)^2} \sum_{s:seq(s)=k} \alpha_t(s)\beta_t(s)$$

Training : the CTC Loss

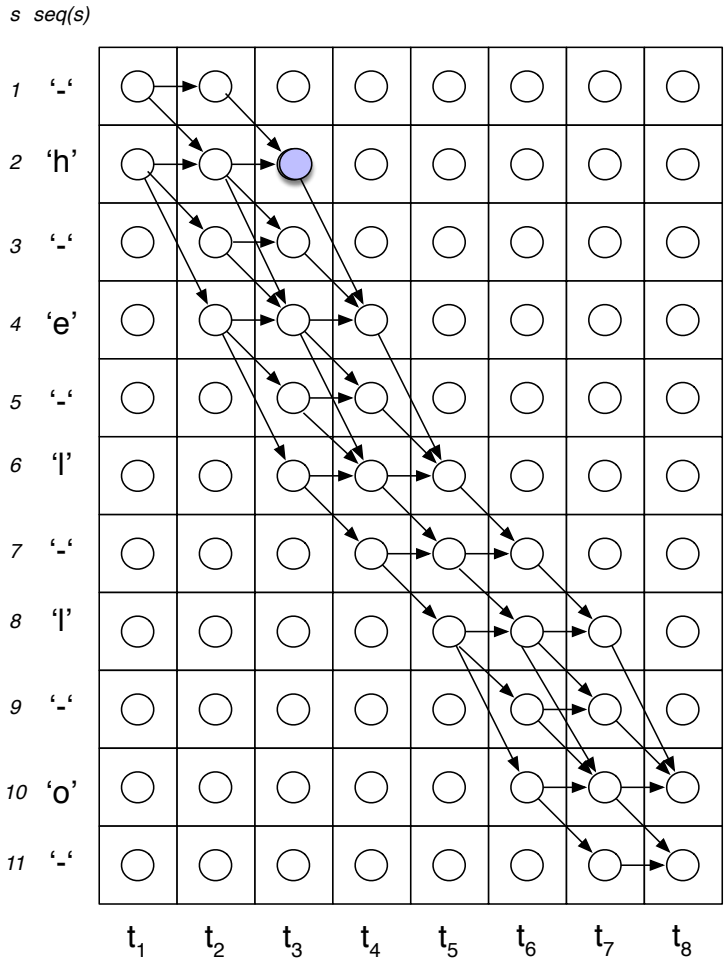
– **Back-propagation example 1**

– Given

- $$\frac{\partial p(''hello'')}{\partial y_t^k} = \frac{-1}{(y_t^k)^2} \sum_{s:seq(s)=k} \alpha_t(s)\beta_t(s)$$

– For $t = 3$ and $k = '' h''$

- h occurs at $s = 2$
- $$\frac{\partial p(''hello'')}{\partial y_{t_3}^{''h''}} = \frac{-1}{(y_{t_3}^{''h''})^2} \alpha_{t_3}(2)\beta_{t_3}(2)$$



Training : the CTC Loss

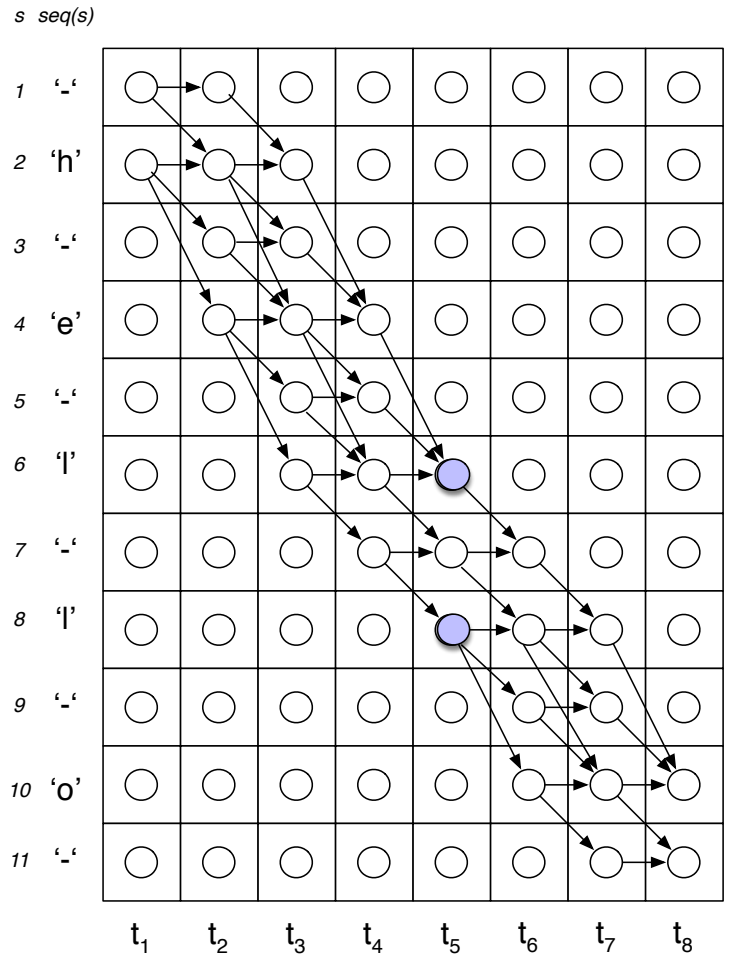
– **Back-propagation example 2**

– Given

- $$\frac{\partial p(''hello'')}{\partial y_t^k} = \frac{-1}{(y_t^k)^2} \sum_{s:seq(s)=k} \alpha_t(s)\beta_t(s)$$

– For $t = 5$ and $k = '' \, l''$

- l occurs at $s = 6$ and $s = 8$
- $$\frac{\partial p(''hello'')}{\partial y_{t_5}^{''l''}} = \frac{-1}{(y_{t_5}^{''h''})^2}[\alpha_{t_5}(6)\beta_{t_5}(6) + \alpha_{t_5}(8)\beta_{t_5}(8)]$$



CTC vs Framewise

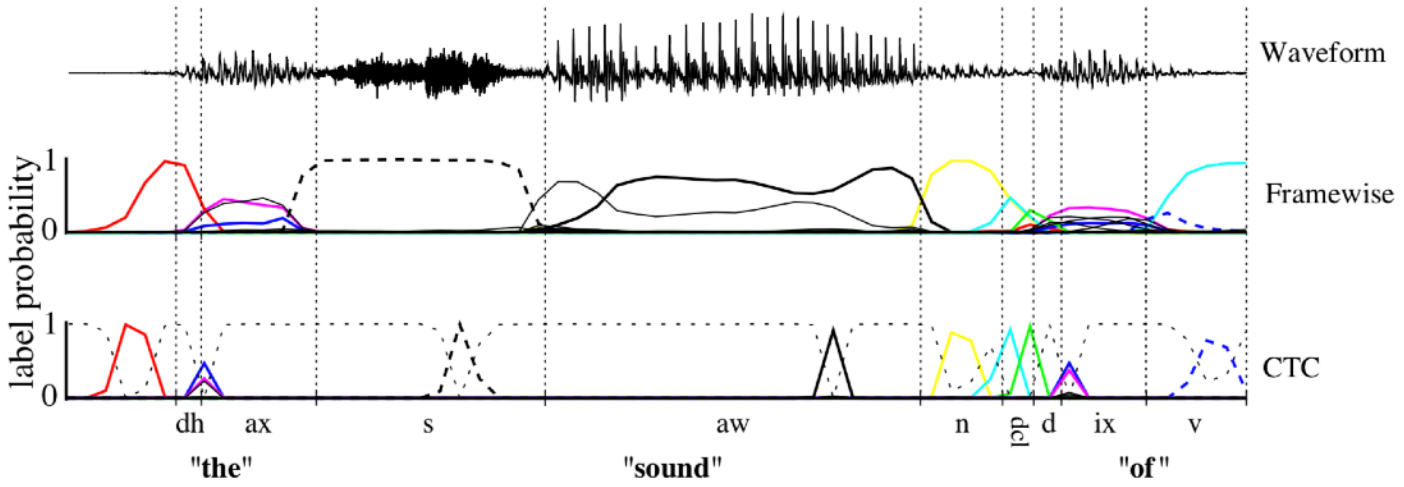


Figure 1. **Framewise and CTC networks classifying a speech signal**. The shaded lines are the output activations, corresponding to the probabilities of observing phonemes at particular times. The CTC network predicts only the sequence of phonemes (typically as a series of spikes, separated by 'blanks', or null predictions), while the framewise network attempts to align them with the manual segmentation (vertical lines). The framewise network receives an error for misaligning the segment boundaries, even if it predicts the correct phoneme (e.g. 'dh'). When one phoneme always occurs beside another (e.g. the closure 'dcl' with the stop 'd'), CTC tends to predict them together in a double spike. The choice of labelling can be read directly from the CTC outputs (follow the spikes), whereas the predictions of the framewise network must be post-processed before use.

Connectionist Temporal Classification

Example of results with CTC

AS OF APRIL FIRST ALL INTEREST INCOME WILL BE TAXED AT TWENTY PERCENT.

BLSTM-CTC transcribed grapheme string:

a s SP o f SP a i p r o l f i r s t SP a l SP i n t e r e s t SP i n c o m e SP w i l l SP b e SP t a x t SP i t SP t w e n t i n SP p e r c e n t SP

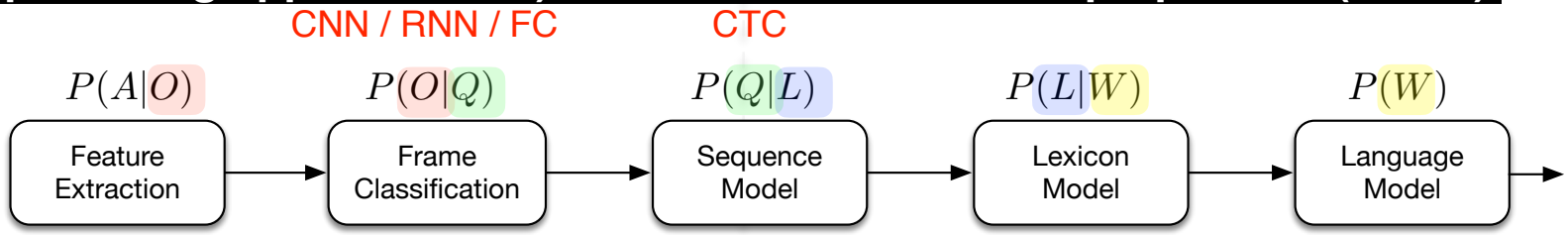
Final output after decoding:

AS OF APRIL FIRST AL **INTEREST INCOME WILL BE** TAX T. IT **TWENTY PERCENT.**

W SJ1 example sentence 1. Average recognition case. Manual annotation (top), BLSTM-CTC grapheme output-SP denotes short-pause, i.e. a word boundary (middle), BLSTM-CTC grapheme output after dictionary based (unweighted, no language model) decoding (bottom), see section III. Correctly recognised words are printed in bold-face.

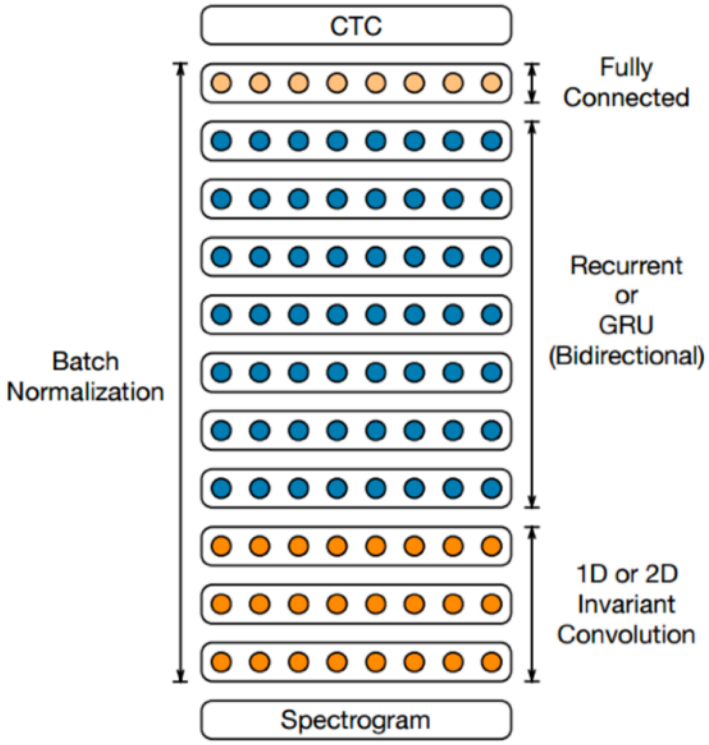
Model	Prediction
Ground Truth	SILENCE
CTC	SILENCE
RNN-Transducer	SILENCE
Attention	i want to get to get to get to get to get to get to get to do that
Ground Truth	play the black eyed peas songs
CTC	
+ Greedy	lading to black **irpen** songs
+ Beam Search + LM	leading to black **european** songs
RNN-Transducer	
+ Greedy	play the black **eye piece** songs
+ Beam Search	play the black **eye piece** songs
+ LM rescore	play the black eyed peas songs
Attention	
+ Greedy	play the black **eyed pea** songs
+ Beam Search	play the black **eyed pea** songs
+ LM rescore	play the black eyed peas songs

Deep learning approaches: 3) End-to-End models: Deep Speech 2 (Baidu)



- 3 layers of 2D-invariant convolution
- 7 layers of bidirectional simple recurrence 100M para
- Word error rates

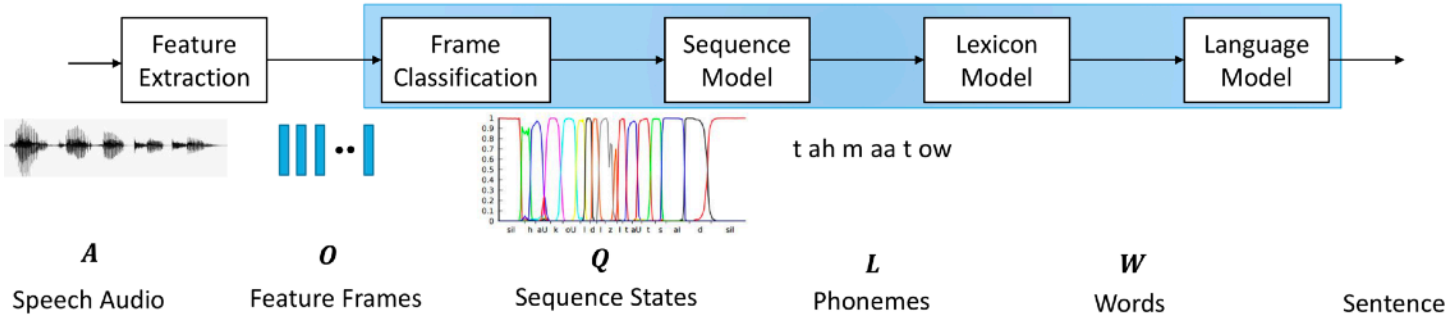
Test set	Deep speech 2	Human
WSJ eval'92	3.60	5.03
WSJ eval'93	4.98	8.08
LibriSpeech test-clean	5.33	5.83
LibriSpeech test-other	13.25	12.69



Example code
Baidu Deep Speech 2

- https://colab.research.google.com/drive/13LmijDMS0EABJjWcN2y_iTyL5BVjo4BI#scrollTo=nCiAaT_WWrh8

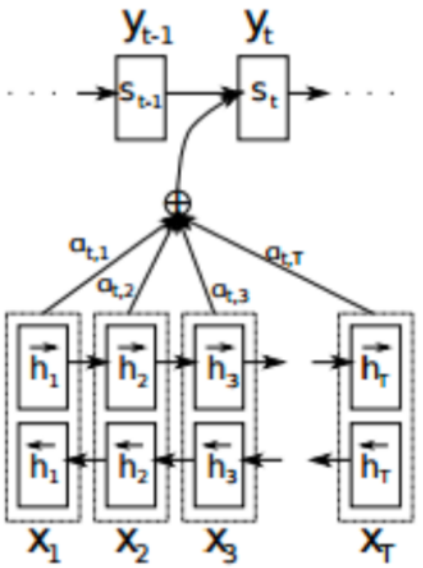
Deep learning approaches: 3) End-to-End models: Attention mechanisms



- **Predict character sequence directly**
- Attention mechanisms also use in
  - Image generation
  - Image text embedding
  - Question answering
  - Machine translation
- Weighted sum of previous history

$$\alpha_{t,j} = softmax(e_{t,j}) = \frac{\exp(e_{t,j})}{\sum_{k=1}^{T} \exp(e_{t,k})}$$

- Essential for modelling long sequences like speech



Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

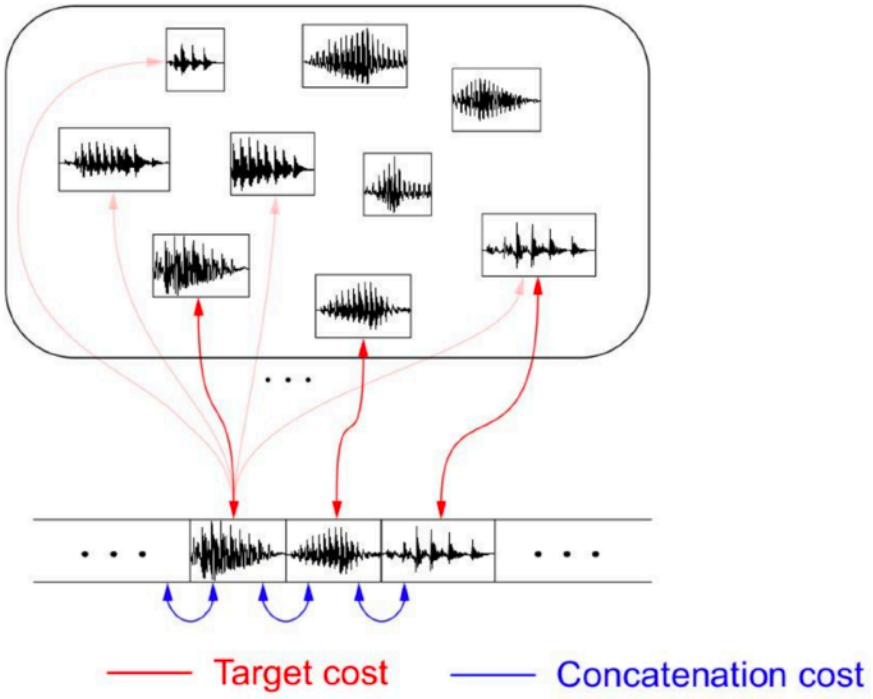
Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

TTS : Text To Speech
Traditional approach : 1) Concatenative synthesis

Concatenative speech synthesis

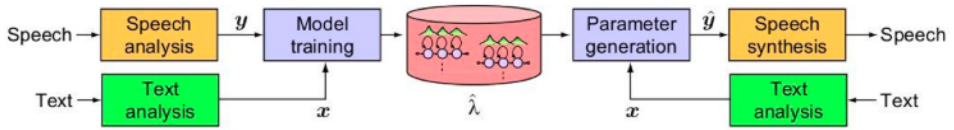
- Unit selection and concatenation
  - Concatenate best pre-recorded speech units
  - Target cost + Concatenation cost
- Criteria :
  - Intelligibility
  - Naturalness

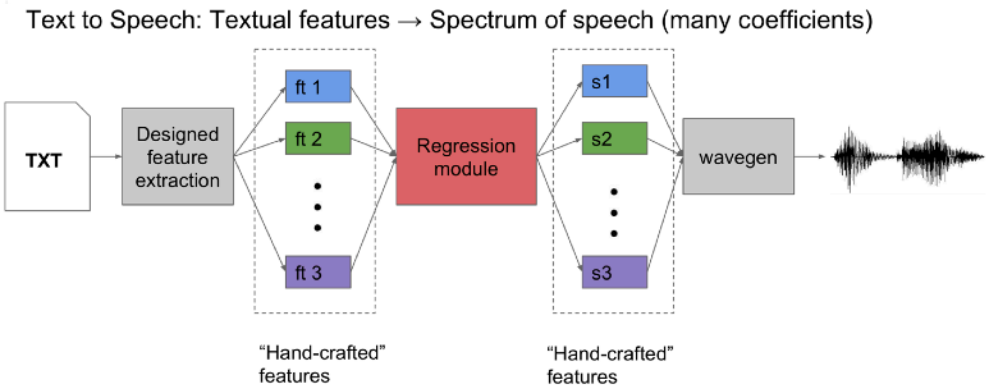


TTS : Text To Speech
Traditional approach : 2) Statistical Speech Synthesis

Statistical Speech Synthesis

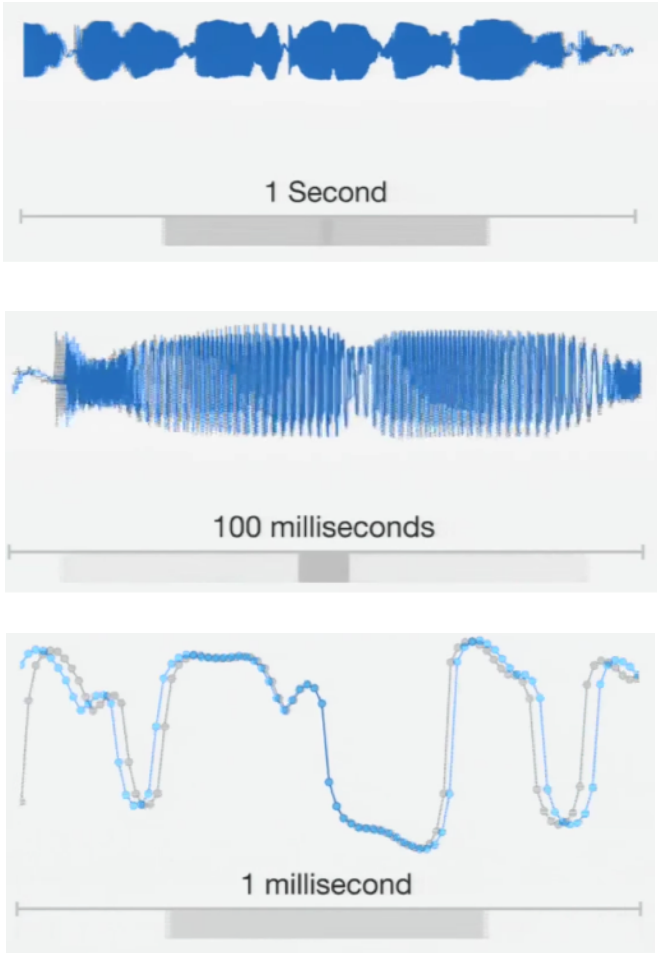
- 1) Represent speech waveform using parameters
  - from text to phoneme (pronunciation)
  - from phoneme to phoneme (with linguistic features)
  - speech features : phoneme duration, fundamental frequency, cepstra
- 2) Use statistic generative model
  - Regression of the parameters using HMM
- 3) Reconstruct waveform from generated parameters
  - vocoder (phase reconstruction)





2016 → WaveNet

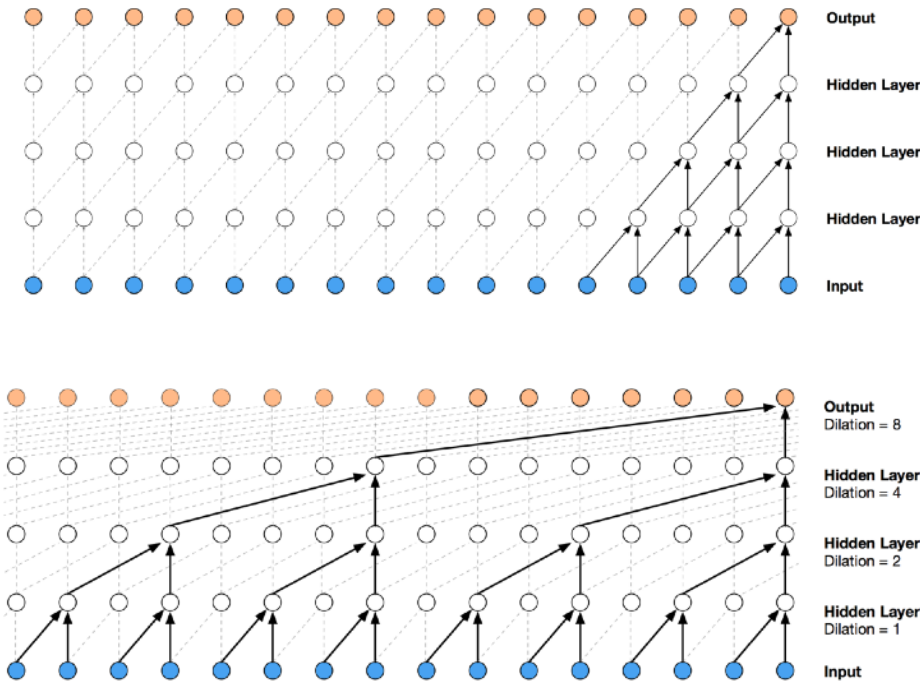
- **Generative model operating directly on audio samples**
  - raw audio = challenging to model because structure at very different scales
  - Based on PixelCNN
  - High resolution signal and long-term dependencies
    - dependencies at **different level**
- **Auto-regressive model**
  - next sample is (almost) reconstructed from **linear** causal combination of past samples
  - $p(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} p(x_t \mid x_1, \ldots, x_{t-1})$
  - $x_t = \sum_{p=1}^{P} \alpha_t x_{t-p} + \epsilon_t$
- **Neural Auto-regressive model:**
  - $x_t = f_\theta(x_1, \ldots, x_{t-1})$



[A. Van Den Oord et al., "**WaveNet : A generative model for raw audio**", 2016] LINK
[S. Mehri et al. "**Sample-RNN: An unconditional end-to-end neural audio generation model**", In ICLR, 2017] LINK

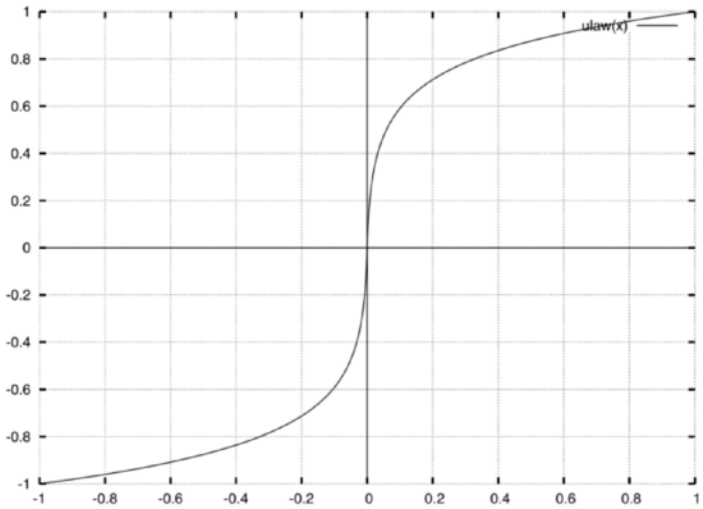
2016 → WaveNet

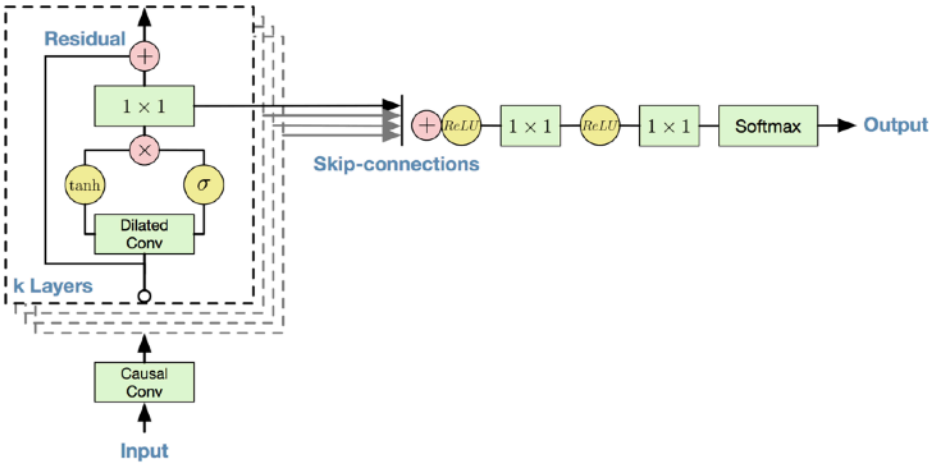
- **Neural Auto-regressive model:**
  - $x_t = f_\theta(x_1, \ldots, x_{t-1})$
  - using RNN ⇒ costly
  - using CNN ⇒ cheap (all convolution can be done in parallel)
- **Causal Convolution**
  - problem : require many layers, or large filters to increase the receptive field.
- **Dilated Convolution** (convolution à trou) :
  - skipping input values with a certain step
  - increase the receptive field by orders of magnitude
- **Stacked dilated convolutions**
  - dilation is doubled for every layer up to a limit and then repeated
    - 1,2,4,...,512,1,2,4,...,512,1,2,4,..., 512.
- Generation is slow :
  - predictions sequential (2 m to generate 1 s)



2016 → WaveNet

- **Input/output signal representation**
  - using μ-law ⇒ 8 bits ⇒ 256 categories
  - $f(x_t) = \text{sign}(x_t)\frac{\log(1+\mu|x_t|)}{\log(1+\mu)}$
    - with $-1 < x_t < 1$ and $\mu = 256$
- **Softmax layer**
  - trained to maximize log-likelihood of the data w.r.t. parameters
- **Gated Activation Units**
  - $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$
  - with
    - $*$ the convolution
    - $\odot$ the element-wise multiplication
- **Residual and Skip connections**
  - to speed up convergence and enable training of much deeper models





- **Conditional Wavenet :** $p(\mathbf{x}\,|\,\mathbf{h}) = \prod_{t=1}^{T} p(x_t\,|\,x_1, \ldots, x_{t-1}, \mathbf{h})$
  - conditioning the model on other input variables $\mathbf{h}$

- **Global conditioning** (speaker ID, music genre, instruments)
  - $\mathbf{h}$: single latent representation
  - influences the output distribution across all time steps,
    - e.g. a speaker embedding in a TTS model
  - $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T \mathbf{h}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k}^T \mathbf{h})$
    - $V_{*,k}$ is a learnable linear projection
    - $V_{*,k}^T \mathbf{h}$ is broadcasted over the time dimension

- **Local conditioning** (text, linguistic features + f0 + duration model)
  - h_t: time series, possibly with a lower sampling frequency than the audio signal,
    - e.g. linguistic features in a TTS model
  - First transform this time series using a transposed convolutional network (learned upsampling)
    - (maps it to a new time series $\mathbf{y} = f(\mathbf{h})$with the same resolution as the audio signal
  - $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T * \mathbf{y}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k}^T * \mathbf{y})$

2016 → WaveNet

– **Perceptual MOS tests**

Speech samples	Subjective 5-scale MOS in naturalness	
	North American English	Mandarin Chinese
LSTM-RNN parametric	3.67 ± 0.098	3.79 ± 0.084
HMM-driven concatenative	3.86 ± 0.137	3.47 ± 0.108
WaveNet (L+F)	$\mathbf{4.21} \pm 0.081$	$\mathbf{4.08} \pm 0.085$
Natural (8-bit μ-law)	4.46 ± 0.067	4.25 ± 0.082
Natural (16-bit linear PCM)	4.55 ± 0.075	4.21 ± 0.071

Table 1: Subjective 5-scale mean opinion scores of speech samples from LSTM-RNN-based statistical parametric, HMM-driven unit selection concatenative, and proposed WaveNet-based speech synthesizers, 8-bit μ-law encoded natural speech, and 16-bit linear pulse-code modulation (PCM) natural speech. WaveNet improved the previous state of the art significantly, reducing the gap between natural speech and best previous model by more than 50%.

– **Audio examples**

- https://deepmind.com/blog/article/wavenet-generative-model-raw-audio

– **Pytorch code**

- https://github.com/vincentherrmann/pytorch-wavenet
- https://github.com/Dankrushen/Wavenet-PyTorch
- https://github.com/ButterscotchV/Wavenet-PyTorch

2016 → WaveNet

Making Music

Since WaveNets can be used to model any audio signal, we thought it would also be fun to try to generate music. Unlike the TTS experiments, we didn't condition the networks on an input sequence telling it what to play (such as a musical score); instead, we simply let it generate whatever it wanted to. When we trained it on a dataset of classical piano music, it produced fascinating samples like the ones below:

0:00 / 0:10

0:00 / 0:10

0:00 / 0:10

0:00 / 0:10

0:00 / 0:10

0:00 / 0:10

https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Example code
WaveNet

- https://colab.research.google.com/drive/1QtsUkwDKgqeAqbD9wvWec8Jxo6eCfJlV

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

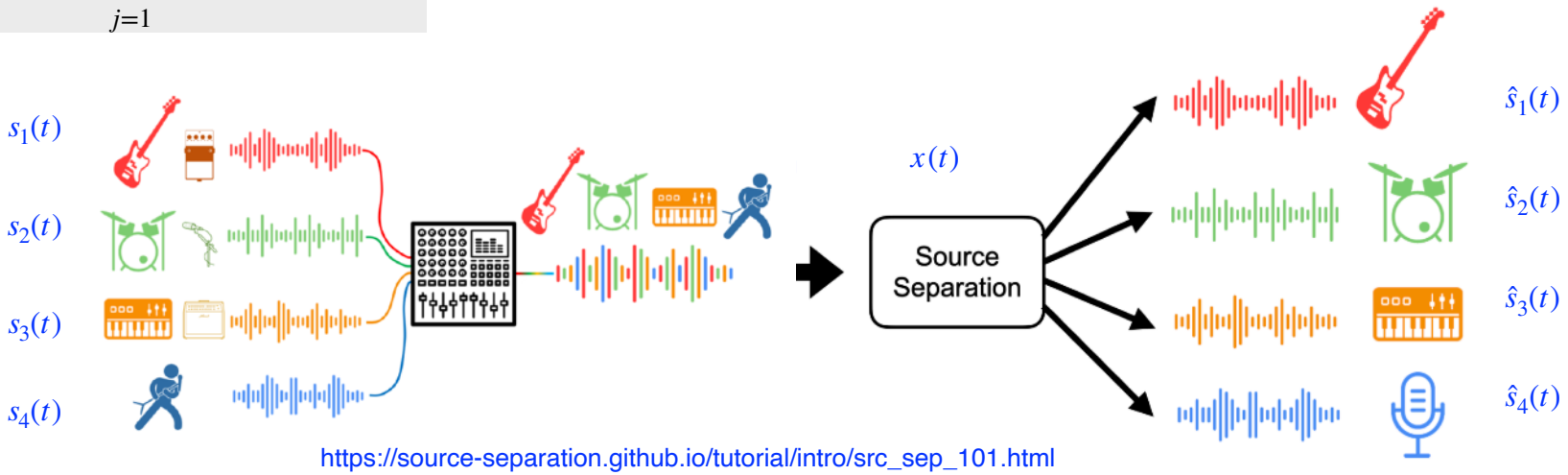
Application: multi-f0 estimation, DDSP

Blind Audio Source Separation (BASS)

Task

– recover one or several source signals $s_j(t)$ from a mixture signal $x(t)$ without any additional information

$$x(t) = \sum_{j=1}^{C} s_j(t) \quad \rightarrow \quad \hat{s}_j(t) \quad ?$$



https://source-separation.github.io/tutorial/intro/src_sep_101.html

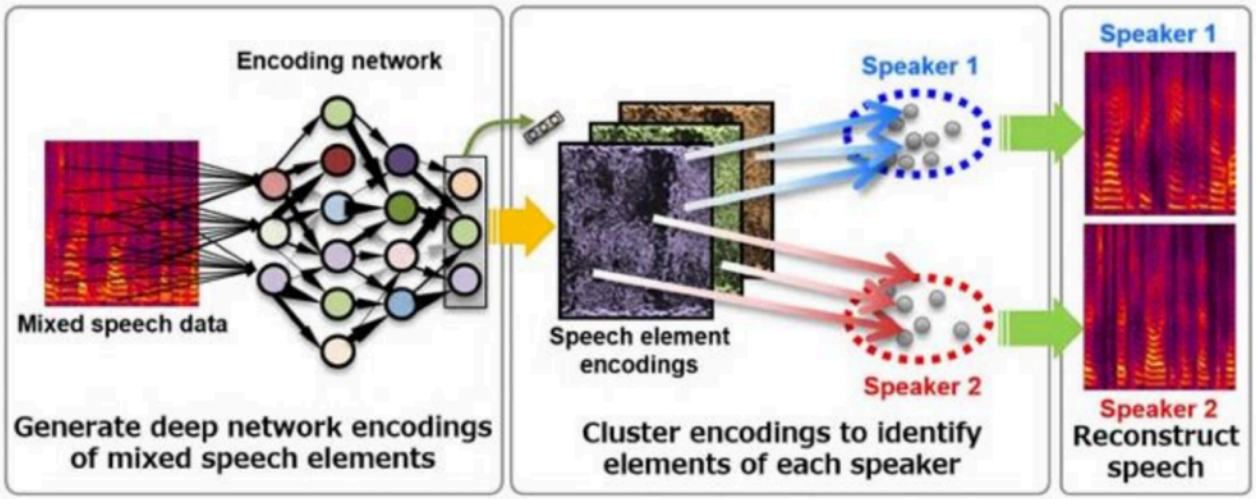
– Closely related to speech denoising, speech enhancement

Applications

– automatic music transcription, lyric and music alignment, musical instrument detection, lyric recognition, automatic singer identification

2016 → Deep Clustering

- Train a DNN (*Bi-LSTM, Bi-LSTM, FC*) to produce **spectrogram embeddings** that are **discriminative** for partition labels given in training data (Class independent)



Source: https://de.mitsubishielectric.com/en/news-events/releases/global/2017/0524-e/index.html

- **Input:** log-STFT $X_{n=(t,f)}$ $n \in \{1,...N\}$
- **Output:** embedding: $V = f_\theta(x) \in \mathbb{R}^{NxK}$,K = size of the projection, we require $\|v_n\|^2 = 1$
- **Ground-Truth:** $Y = \{y_{n,c}\}, y_{n,c} = 1$ if n in partition c
  - → Ideal pairwise affinity matrix: YY^T / Estimated pairwise affinity matrix: VV^T
- **Training**: minimize $C_\theta = |VV^T - YY^T|^2_W$
- **Inference ?** clustering (k-means) of the $n = (t, f)$ embedding points $V = f_\theta(x)$

[J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe. "**Deep clustering: Discriminative embeddings for ...**", In IEEE ICASSP, 2016] LINK

Audio Source Separation - Systems
General Source Separation Architecture

- **Encoder:**
  - Fixed filterbanks (STFT, Mel)
  - Trainable filterbanks
  - Free filter-banks
- **Separator**
  - LSTMs, TCN, U-Net, ...
  - Complex Network
- **Decoder**
  - Inverse encoders ($STFT^{-1}$)
  - Neural vocoder
- **Loss**
  - Fixed or Permutation invariant loss (PIT)
  - Spectrogram loss, l1, l2, SI-SNR

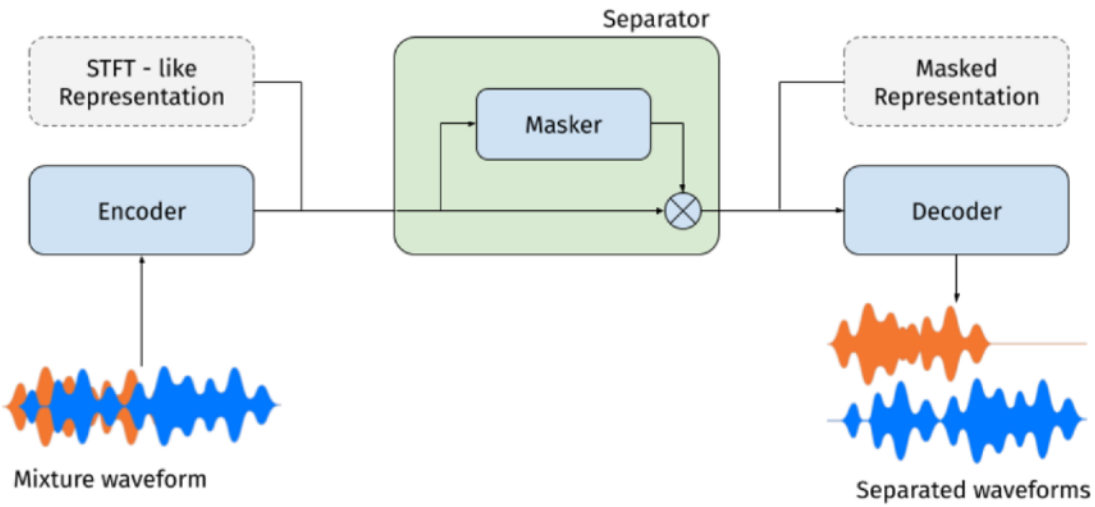
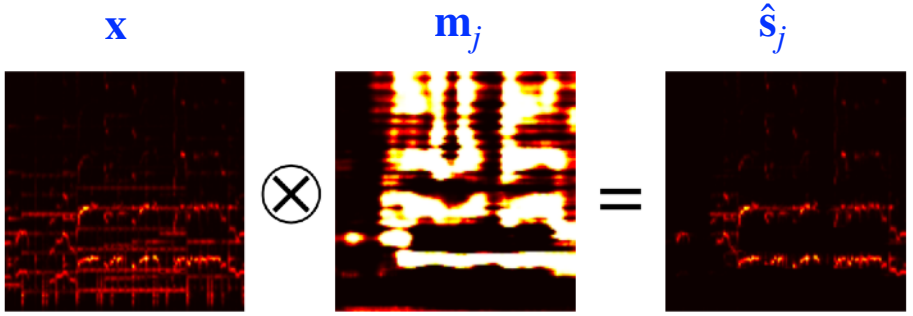


Figure from Pariente, Manuel, et al. "Asteroid: the PyTorch-based audio source separation toolkit for researchers." *arXiv preprint arXiv:2005.04132* (2020).

Audio Source Separation
Masks ?

- Given a mixture $x(t)$ of C sources $s_i(t)$:
  - $x(t) = \sum_{j=1}^{C} s_j(t) \Leftrightarrow X(t,f) = \sum_{j=1}^{C} S_j(t,f)$
  - Estimated magnitude STFT of source j
    - notation: $\mathbf{x} = |X(t,f)|$
    - $\hat{\mathbf{s}}_j = \mathbf{x} \odot \mathbf{m}_j$ subject to $\sum_{j=1}^{C} \mathbf{m}_j = 1$



  - **IBM (Ideal Binary Mask)** for source j is defined as
    - $IBM_{j,tf} = \delta(|\mathbf{s}_{j,tf}| > |\mathbf{s}_{j',tf}|), \forall j' \neq j \quad \in \{0,1\}$
      - i,e. source j it is the most dominant source for this specific STFT bin (t,f)
  - **IRM (Ideal Ratio/Soft Mask)** for source j is defined as
    - $IRM_{j,tf} = \frac{|\mathbf{s}_{j,tf}|}{\sum_{j'=1}^{C} |\mathbf{s}_{j',tf}|} \in [0,1]$
  - **Wiener-filter Like Mask (WFM)** for source j is defined as
    - $WFM_{j,tf} = \frac{|\mathbf{s}_{j,tf}|^2}{\sum_{j'=1}^{C} |\mathbf{s}_{j',tf}|^2}$

https://arxiv.org/pdf/1707.03634.pdf

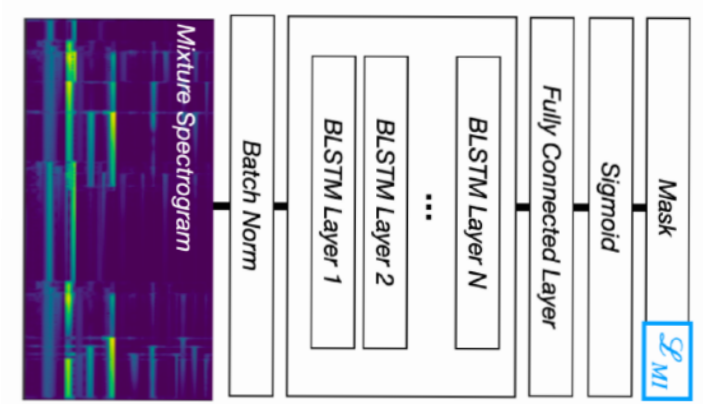
Audio Source Separation Systems

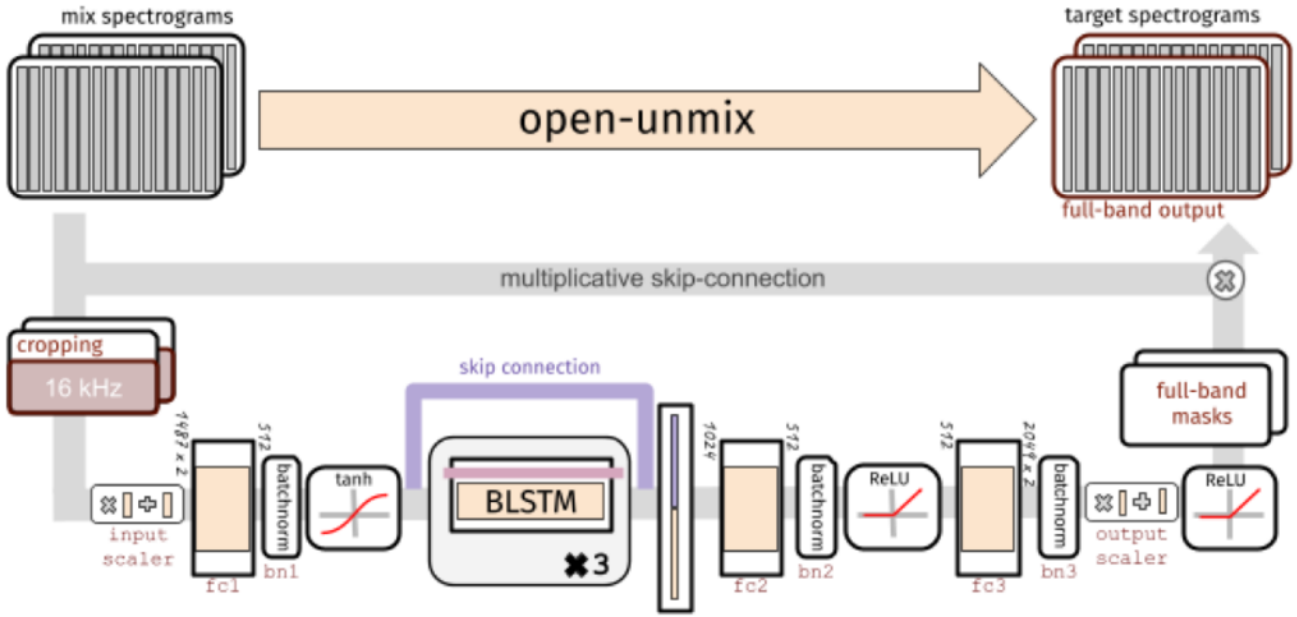
Using STFT as Input

2019 → Open-Unmix using BLSTM

Idea:

- predict the soft mask using LSTMs
- add skip-connection around LSTMs
- before feeding the LSTMs, first reduce the frequency dimension of the |STFT| using FC



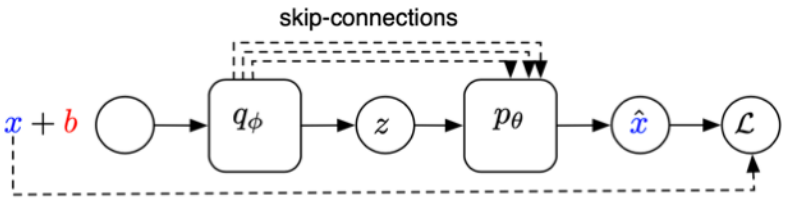


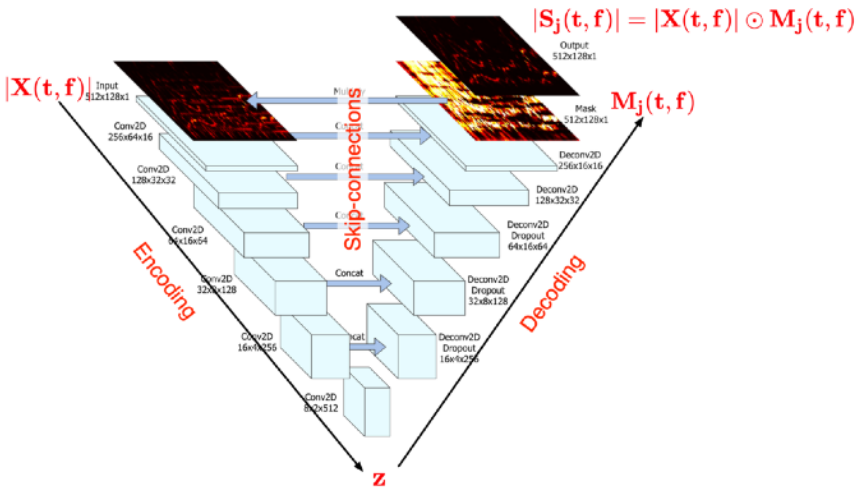
F.-R. Stöter et al, "**Open-Unmix - A Reference Implementation for Music Source Separation**", Journal of Open Source Software, 2019 LINK

Audio Source Separation - Systems
2017 → using CDAE with skip-connection (U-Net)

Idea:

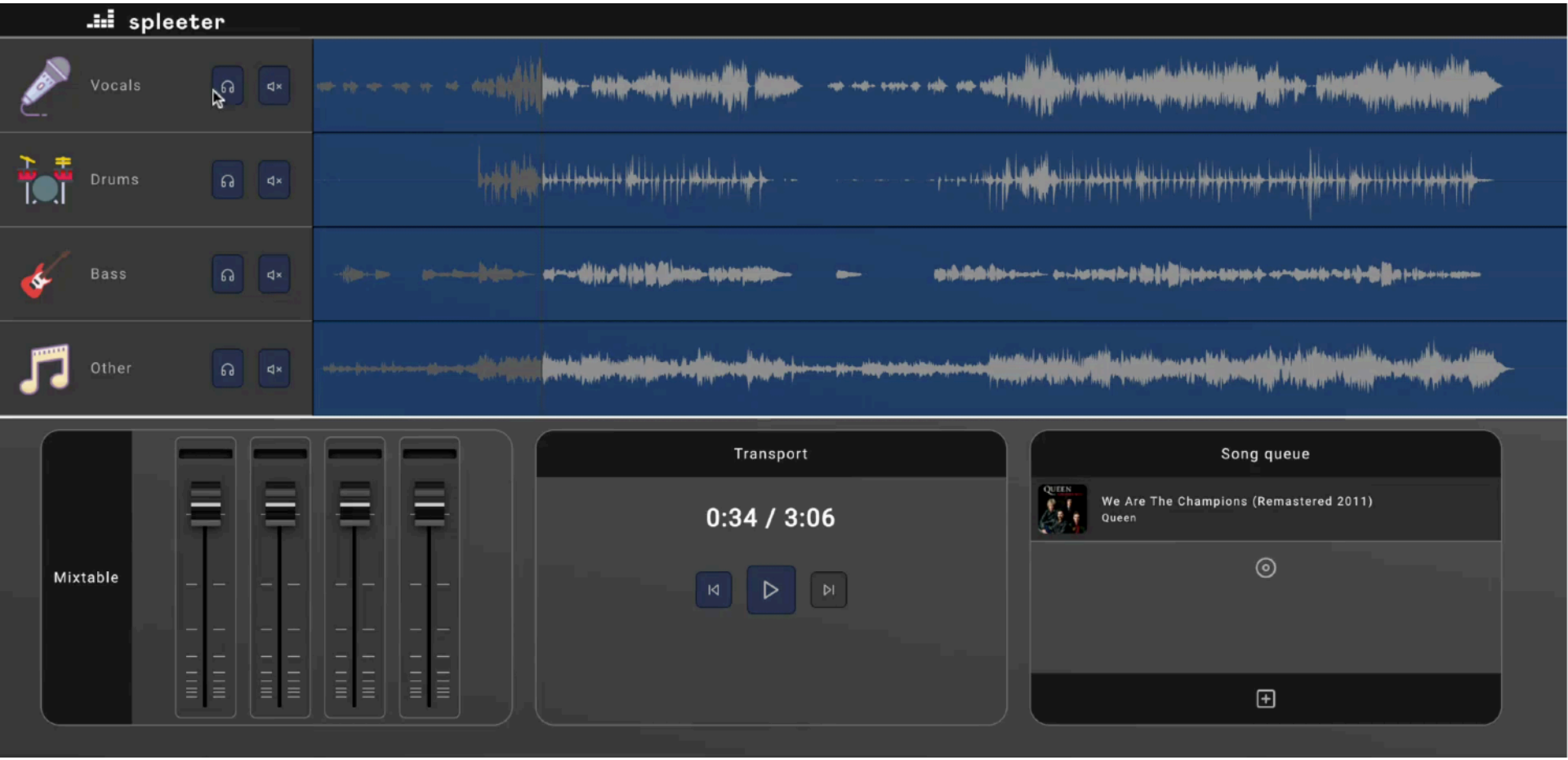
- **Limitations of Convolutional Denoising Auto-Encoder (CDAE):**
  - allows to reconstruct a clean but blurred signal from its noisy version
  - but the fine details of the spectrogram (precise harmonic locations) are lost in z
- **U-Net architecture**
  - an AE with skip-connections between E and D
  - contracting path: capture context and a symmetric expanding path that enables precise localization
- **U-Net for source separation**
  - **Goal**: isolate the singing voice from real polyphonic music
  - **Input**: patches of spectrogram representation $|X(t,f)|$ to
  - **Output**: Time/Frequency mask $M_j(t,f)$
  - **Separation**: apply $M_j(t,f)$ to the amplitude STFT of the mixture $|X(t,f)|$ to separate the amplitude STFT of the isolated source
    - $|S_j(t,f)| = |X(t,f)| \odot M_j(t,f)$





[O. Ronneberger et al. "**U-net:Convolutional networks for biomedical image segmentation**", 2015] LINK
[A. Jansson et al. "**Singing voice separation with deep u-net convolutional networks**". In ISMIR, 2017] LINK

Audio Source Separation - Systems
U-Net → Spleeter



[R. Hennequin et al. "**Spleeter: A fast and state-of-the art music source separation**" in LBD-ISMIR 2019] LINK
https://github.com/deezer/spleeter

Example code
U-Net

- https://colab.research.google.com/drive/1UB9jMUSJ9mrGN28rMxm1QVP0-WPhN7Ch

Audio Source Separation Systems

Using Waveform as Input

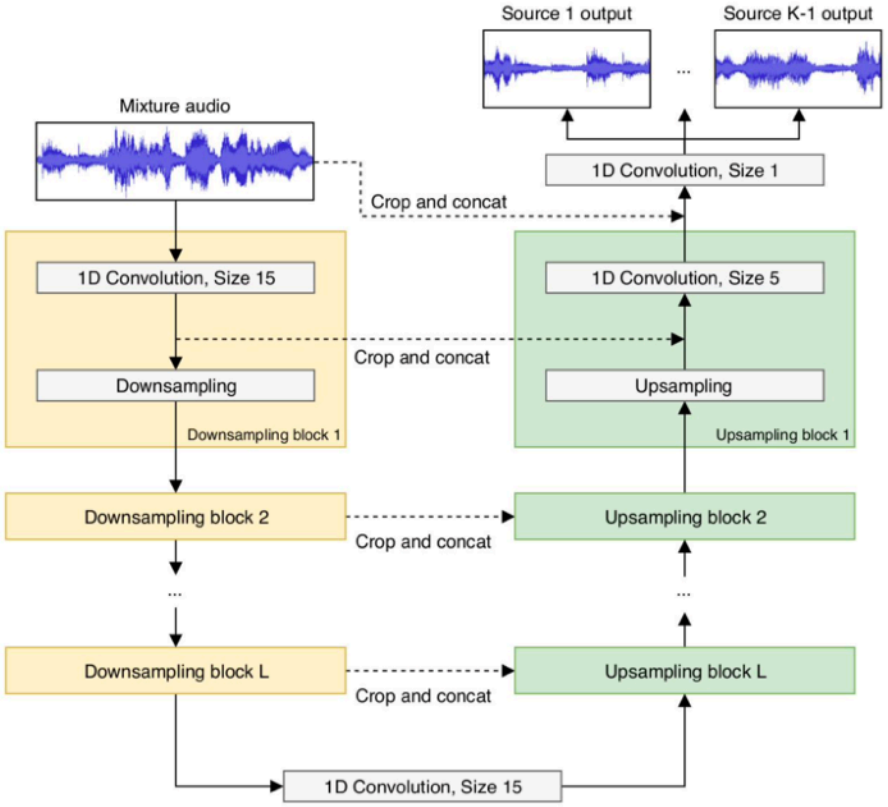
Audio Source Separation - Systems Using Waveform as Input

- **Problems of working in the |STFT| domain**
  - the mask is only applied to |STFT|
  - the audio signal is reconstructed from STFT using original phase
    - not possible to detect/correct potential positive/negative phase interferences
  - using STFT requires choosing analysis parameters (window size, hop size, ...)

- **Working directly in the waveform domain**
  - It is then possible to surpass the Ideal Binary Mask !
  - Two possibilities
    - (a) estimate directly the waveform of separated source
    - (b) estimate mask to be applied to encoded waveform

Audio Source Separation - Systems 2018 → Wave-U-Net

- **Wave-U-Net**
  - estimate directly the waveforms of the separated sources
  - several simultaneous outputs
  - to avoid artifacts:
    - replace *strided-transposed-convolution* (as used in previous works) by upsampling feature maps with *linear interpolation followed by a normal convolution*



[Stoller et al., 2018. **"Wave-u-net: A multi-scale neural network for end-to-end audio source separation"**, in ISMIR, 2018] LINK

Audio Source Separation - Systems
2018 → Tas-Net

Idea :

- reformulate the system as an encoder/masker/decoder
- use LSTM for masker

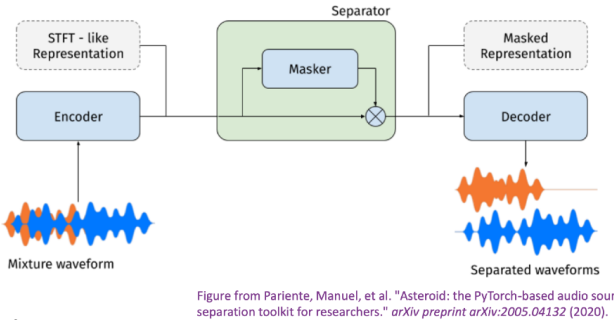
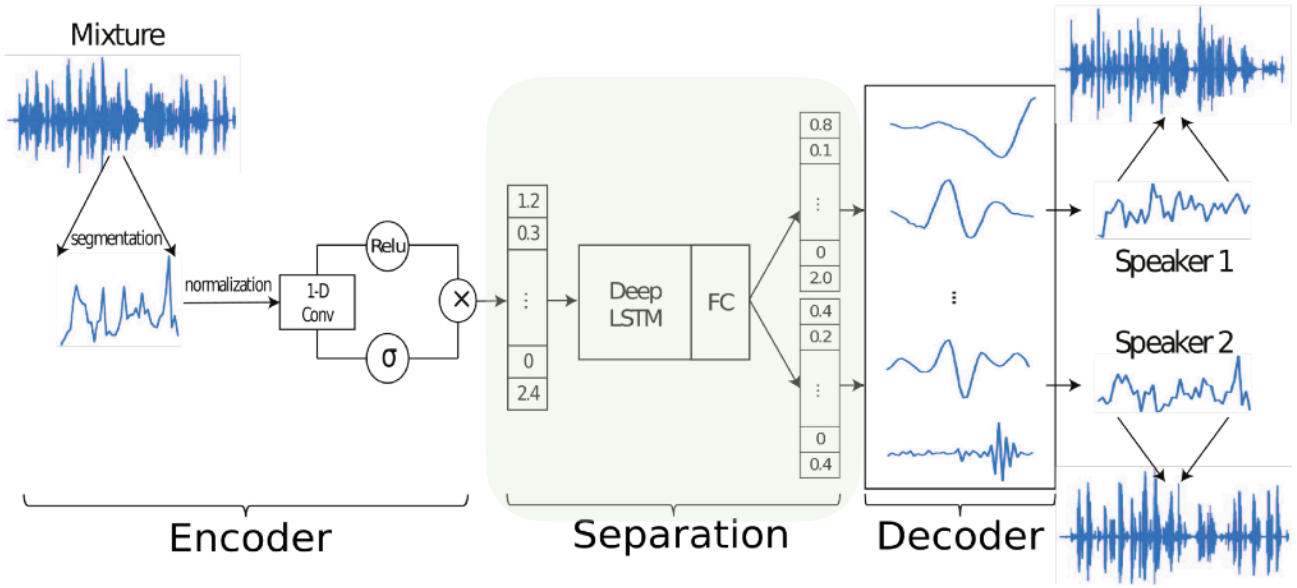


Figure from Pariente, Manuel, et al. "Asteroid: the PyTorch-based audio source separation toolkit for researchers." *arXiv preprint arXiv:2005.04132* (2020).

- **TasNet**: **T**ime-domain **A**udio **S**eparation network
  - **Encoder**: Learn a projection of the audio waveform
  - **Separation**: Learn the mask to be applied to the projection
  - **Decoder**: regenerate the separated audio from the masked projection



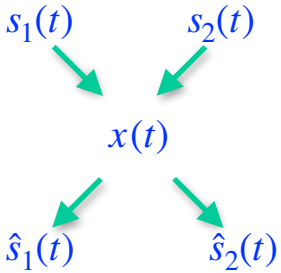
[Y. Luo and N. Mesgarani "**TasNet: time-domain audio separation network**" in ICASSP 2018.] LINK

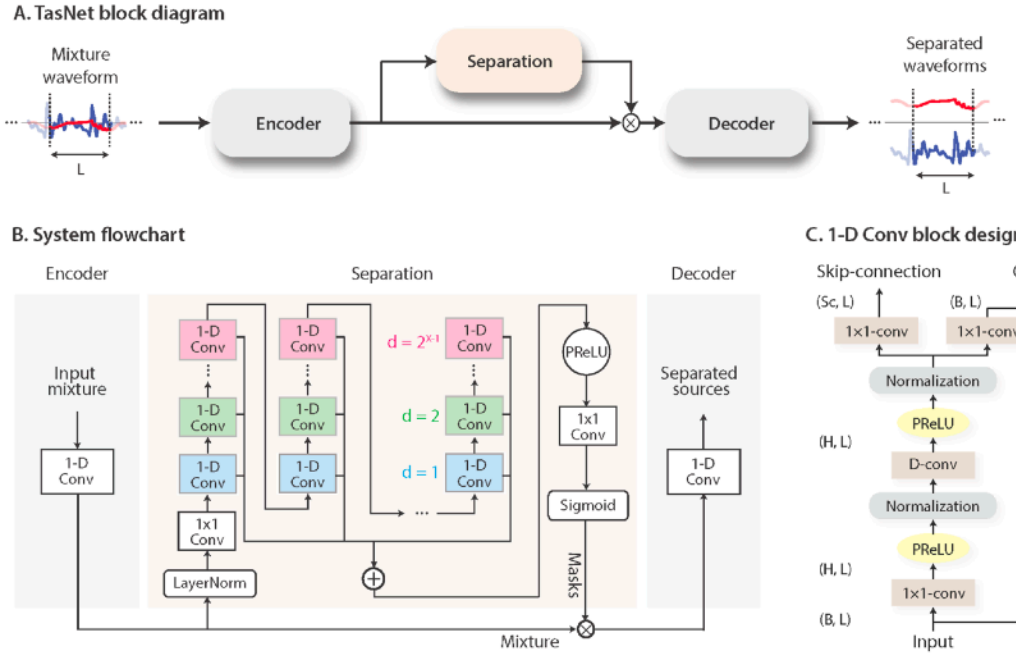
Audio Source Separation - Systems
2018 → Conv-Tas-Net

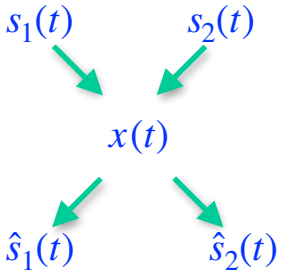
Idea:

- Fully convolutional

- **ConvTasNet**: **T**ime-domain **A**udio **S**eparation network
  - **Encoder**: Learn a projection of the audio waveform
  - **Separation**: Learn the mask to be applied to the projection
  - **Decoder**: regenerate the separated audio from the masked projection

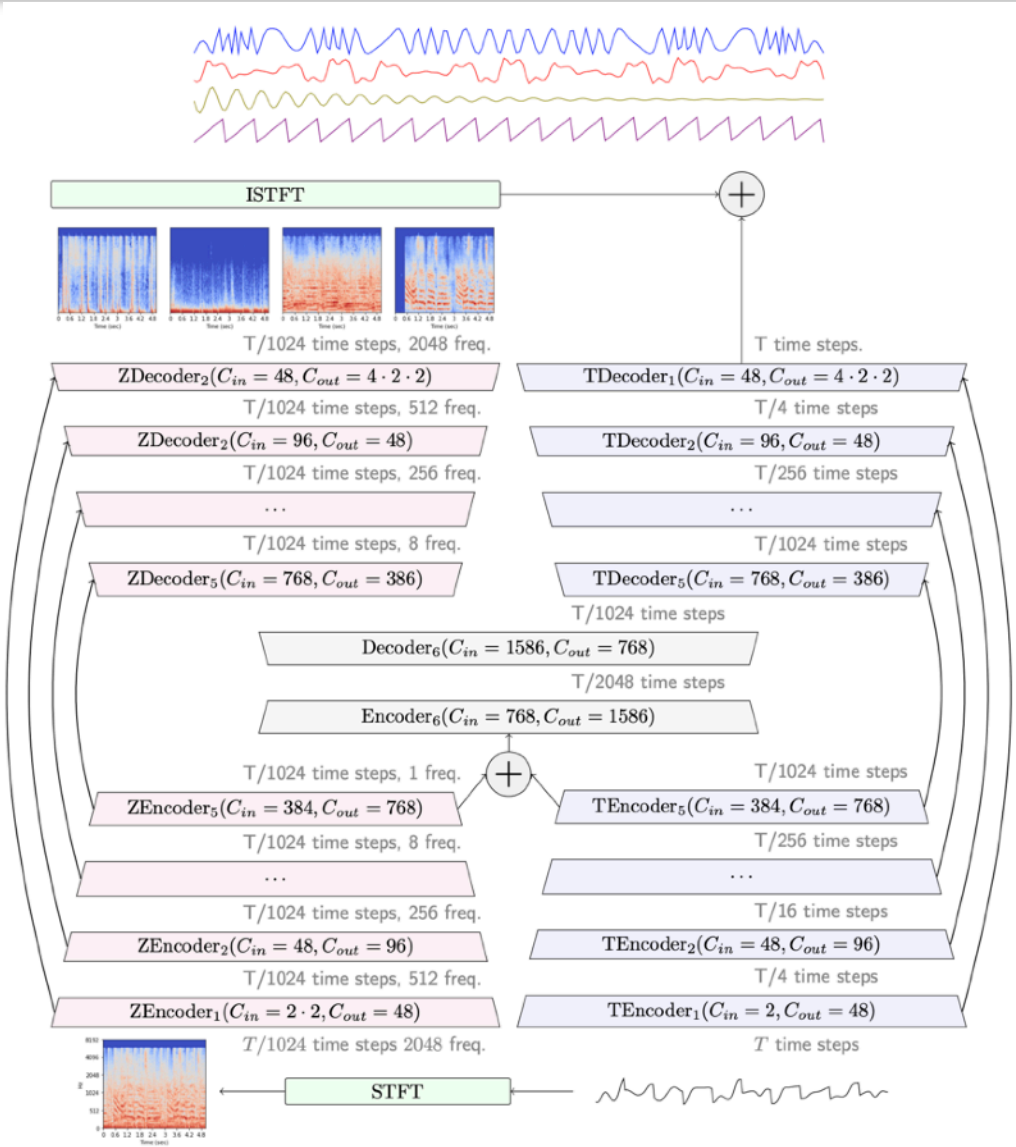






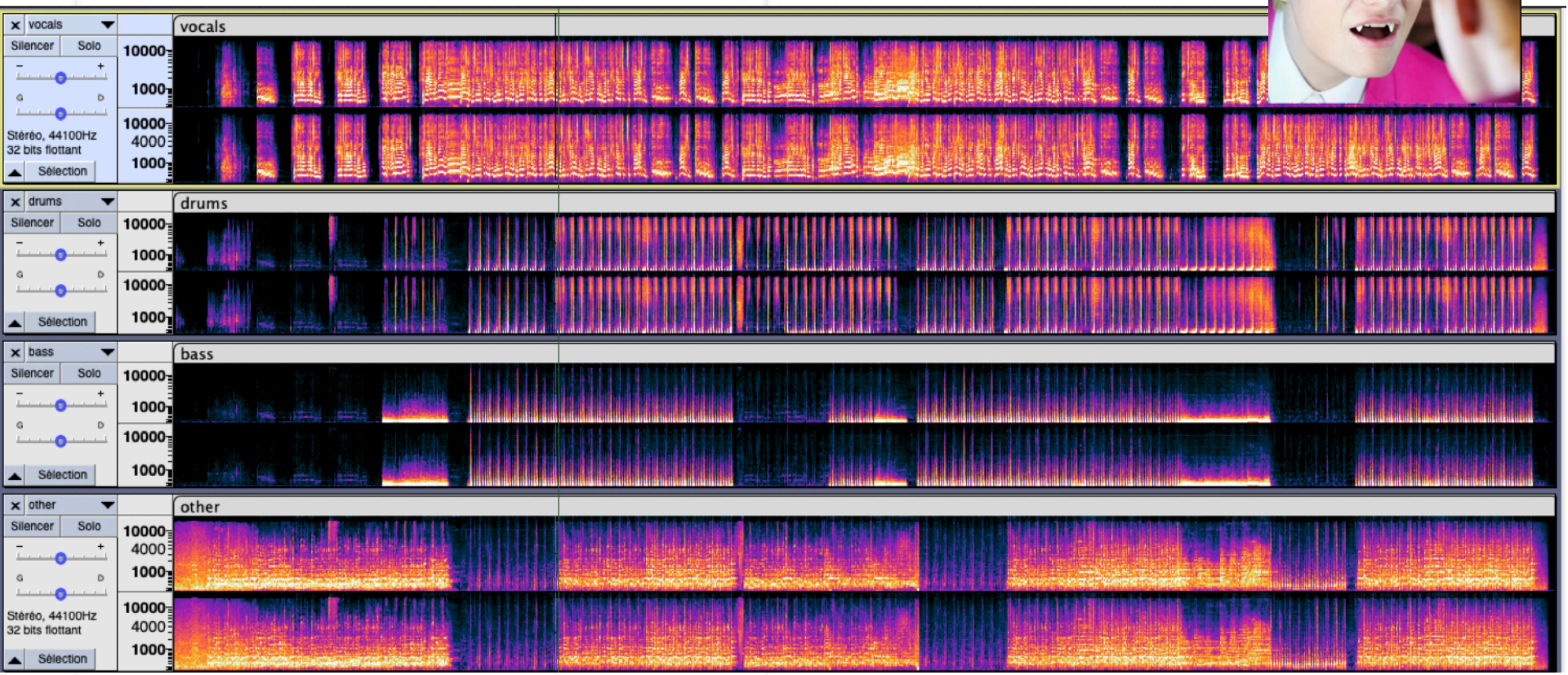
[Y. Luo and N. Mesgarani. "**Conv-TasNet: Surpassing ideal time–frequency ...**" IEEE TASLP, 2019] LINK

Audio Source Separation - Systems
2021 → Demucs 2



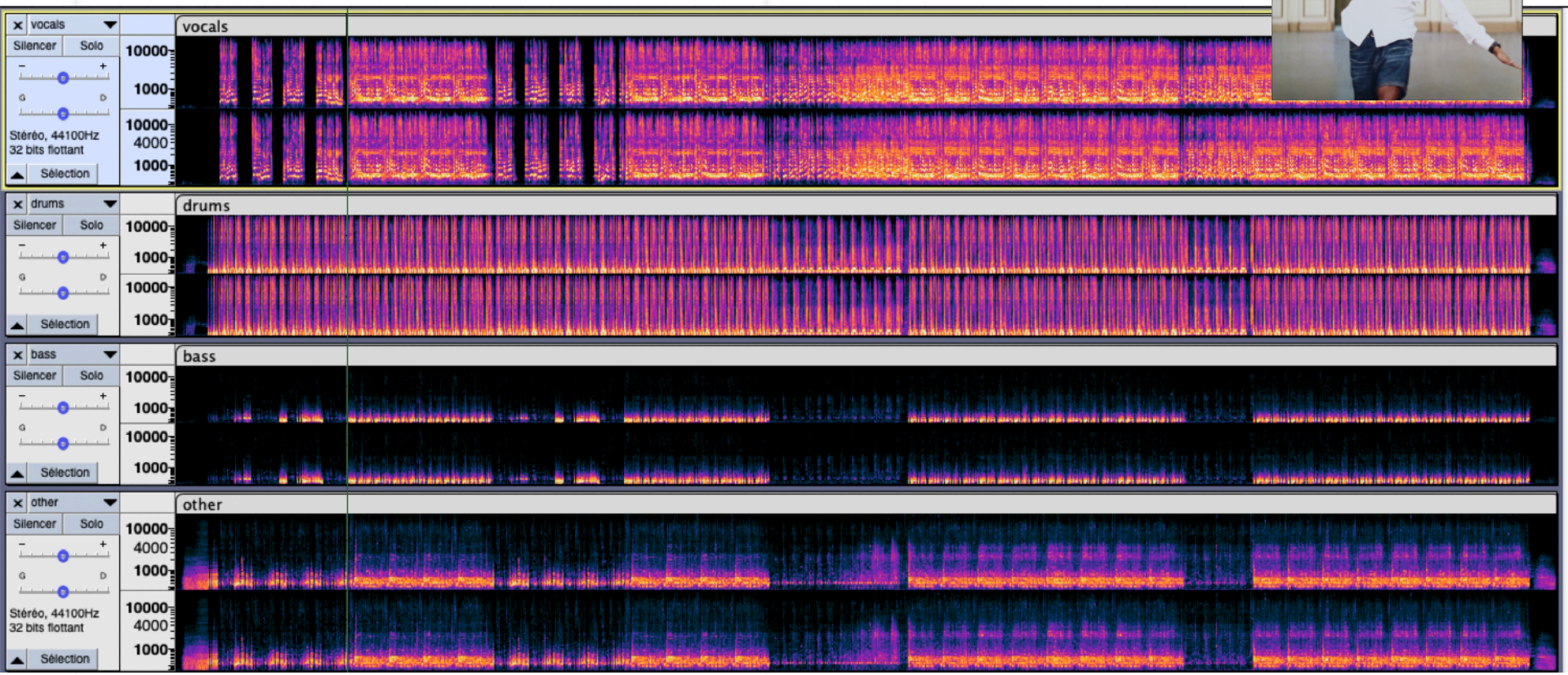
[A. Défossez,"**Hybrid Spectrogram and Waveform Source Separation**", 2021.] LINK

Audio Source Separation - Systems
2021 → Demucs 2



[A. Défossez,"**Hybrid Spectrogram and Waveform Source Separation**", 2021.] LINK

Audio Source Separation - Systems
2021 → Demucs 2



[A. Défossez,"**Hybrid Spectrogram and Waveform Source Separation**", 2021.] LINK

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

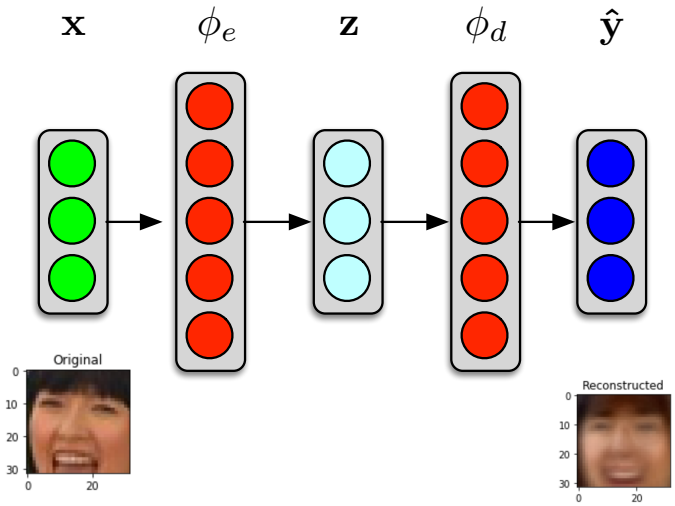
Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

Auto-Encoder (AE)

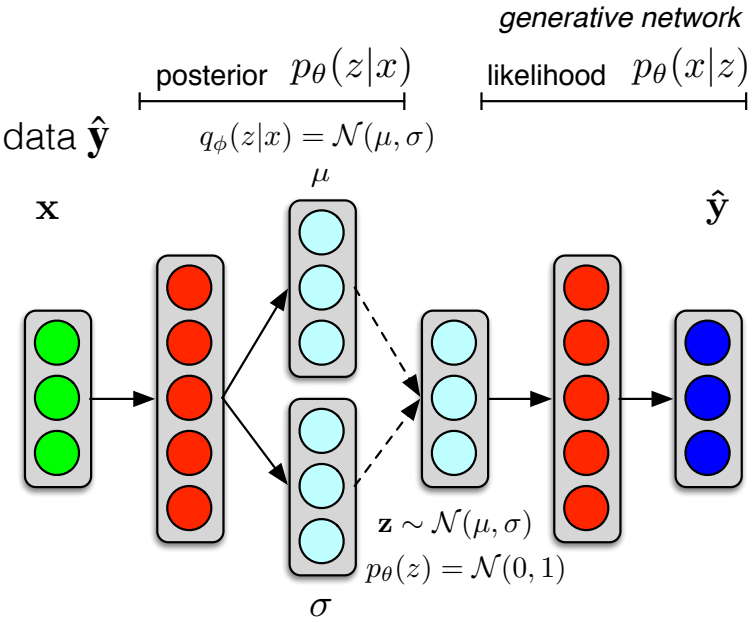
- Two sub-networks:
  - **Encoder** ϕ_e
    - projects the input data $\mathbf{x} \in \mathbb{R}^M$ in a latent representation: $\mathbf{z} = \phi_e(\mathbf{x}) \in \mathbb{R}^d \; (d << M)$
  - **Decoder** ϕ_d
    - attempts to reconstruct the input data from the latent representation $\hat{y} = \phi_d(\mathbf{z})$
  - ϕ_e and ϕ_d can be any type of network (MLP, CNN, RNN, ...)
- **Training**:
  - minimizing a MSE $\arg\min_{\phi_e,\phi_d} \|\mathbf{x} - (\phi_d \circ \phi_e(\mathbf{x}))\|^2$
- Often used for feature learning, compression, ...

- **Variations:**
  - Denoising AE
  - Sparse AE
  - Contractive AE



Variational Auto-Encoder (VAE)

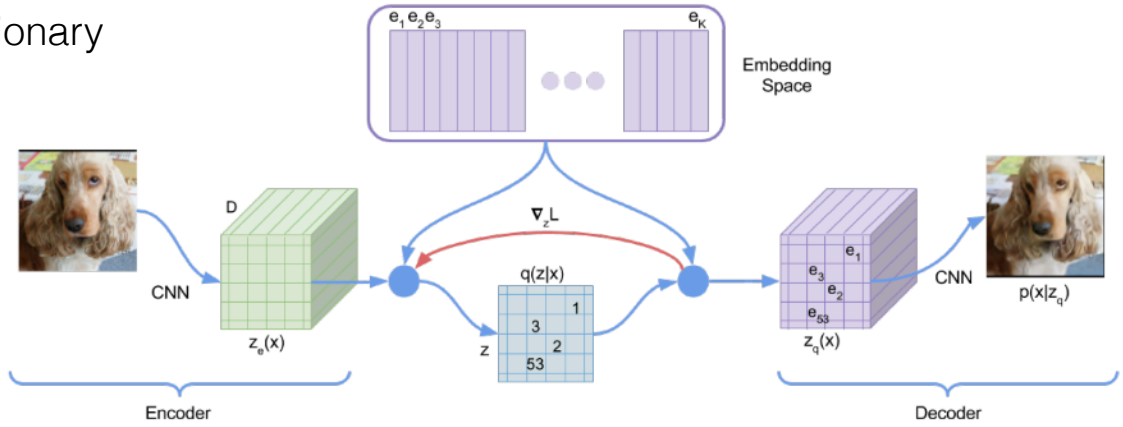
- Generative model:
  - sample points $\mathbf{z}$ in the latent space to generate new data $\hat{\mathbf{y}}$
  - most popular form of AE for generation
- **Encoder**
  - models the posterior $p_\theta(\mathbf{z}\,|\,\mathbf{x})$
- **Decoder** (generative network)
  - models the likelihood $p_\theta(\mathbf{x}\,|\,\mathbf{z})$
- Problem:
  - $p_\theta(\mathbf{z}\,|\,\mathbf{x})$ is intractable,
- Solution:
  - approximate it with $q_\phi(\mathbf{z}\,|\,\mathbf{x})$ (variational Bayesian approach) which is set to a Gaussian distribution $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ (outputs of the encoder)
- **Training**:
  - minimize the KL-divergence between $q_\phi(\mathbf{z}\,|\,\mathbf{x})$ and $p_\theta(\mathbf{z}\,|\,\mathbf{x})$ equivalent to maximize an ELBO criteria
    - need a prior $p_\theta(\mathbf{z})$ which is set to $\mathcal{N}(0,1)$
  - maximize $\mathbb{E}_q[\log(x\,|\,z)]$ using Monte-Carlo ($z \sim q_\phi(z\,|\,x)$)



[Diederik P Kingma and Max Welling. "**Auto-encoding variational bayes",** ICRL, 2014] LINK

VQ-VAE

- Encoder:
  - $z_e(x) \in \mathbb{R}^d$
- Embedding space, codebook, dictionary
  - $e_i \in \mathbb{R}^D, i \in 1,2,\ldots K$
- Quantification:
  - $k = \arg\min_j ||z_e(x) - e_j||_2$
- Input to the decoder:
  - $z_q(x) = e_k$

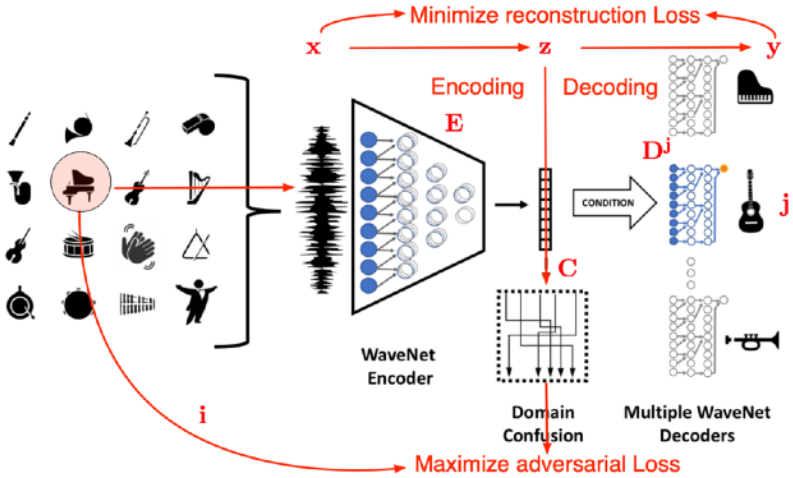


- **Training**:
  - copy gradients $\Delta_z \mathcal{L}$ from decoder input $z_q(x)$ to encoder output $z_e(x)$
- **Total loss**:
  - $\mathcal{L} = \overbrace{\log(p(x \mid z_q(x))}^{reconstruction} + \overbrace{\|sg[z_e(x)] - e\|_2^2}^{VQ} + \beta \overbrace{\|z_e(x) - sg[e]\|_2^2}^{commitment}$
  - **reconstruction**: optimises both the encoder and decoder
  - **VQ**: learn the embedding space, move the embedding vectors e_i towards the encoder outputs $z_e(x)$
  - **Commitment**: to make sure the encoder commits to an embedding and its output does not grow

[A. van den Oord et al. "**Neural discrete representation learning",** NeurIPS, 2017]. LINK

Music translation

- **Task**:
  - translating
    - an input music x with {musical instruments, genres, and style} i to
    - an output y with {musical instruments, genres, and style} j
  - while preserving the musical score
  - without explicitly extracting the musical score and with a single but smart network
- Network has the form of an AE
  - Encoder E (a single WaveNet for all i) is used to project x in the latent space z
  - z is then used to reconstruct an output music track with {musical instruments, genres, and style} j using specific decoders D^j for each j (which are all WaveNet decoders)
- **Training:**
  - Encoder and decoder are trained to minimize a reconstruction loss between x and y.



https://research.fb.com/publications/a-universal-music-translation-network/

[Mor, N.,Wolf, L., Polyak, A., Taigman, Y, "**A universal music translation network",** ICLR, 2019]

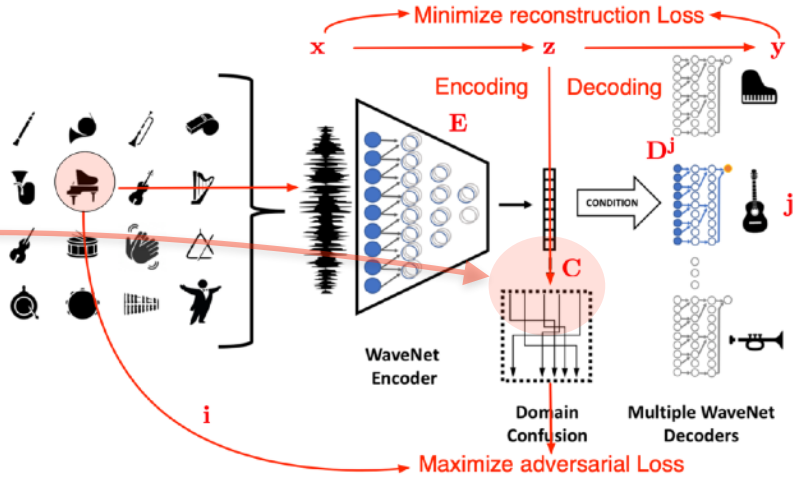
Music translation (cont.)

– **Disentanglement of z ?**

- Important that z does not contain information related to the {musical instruments, genres, and style} i

– How ?

- 1) a classifier C is trained to recognize i from z
- 2) with C fixed, we add an adversarial loss, i.e. we train the encoder E to maximize a classification loss (guarantee that not possible to recognize i from z)



https://research.fb.com/publications/a-universal-music-translation-network/

– **Training:**

- minimize the reconstruction loss and maximize domain classification loss (adversarial loss)
  - prevent z to learn domain characteristic maximize the adversarial loss

$$\sum_j \sum_{s_j} \mathbb{E}_r \mathcal{L}(D^j(E(O(s^j, r))), s^j) - \lambda \mathcal{L}(C(E(O(s^j, r))), j)$$

[Mor, N.,Wolf, L., Polyak, A., Taigman, Y, "**A universal music translation network",** ICLR, 2019]

2019 → Universal music translation

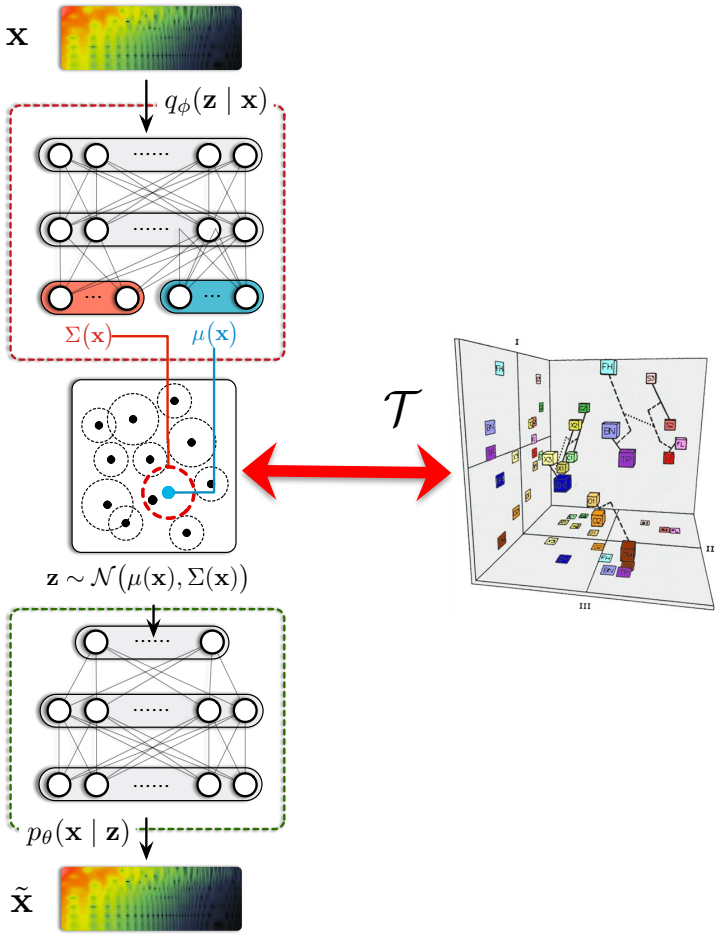
Original (Input): String Quartet, Haydn

- **Encoder/Decoder (Auto-encoder, VAE)**

  - **Regularize the latent spaces to match perceptual distances from timbre studies**

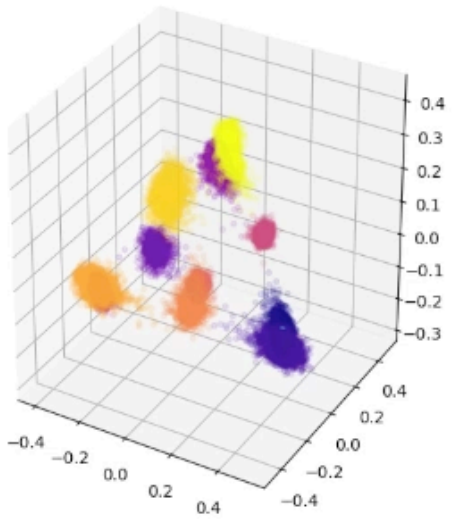
$$\mathbb{E}_q[\log p(x|z)] - \beta \cdot KL(q_\phi(z|x)||p(z)) + \alpha \cdot \mathscr{R}(z, \mathscr{T})$$

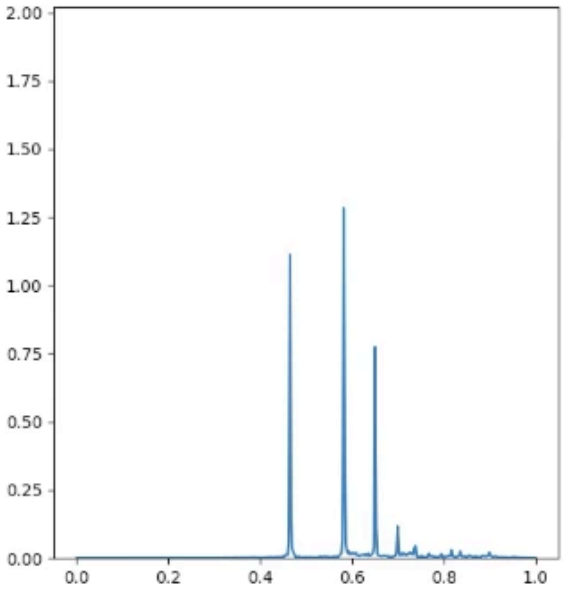
    - we want the distances between instruments in the latent space z to follow the perceptual distance
      - $\mathscr{R}(z, \mathscr{T}) = \sum_{i \neq j} ||D^z_{i,j} - D^{\mathscr{T}}_{i,j}||^2$
      - $D^{\mathscr{T}}_{i,j}$= perceptual ratings of similarity/ dissimilarity



[P. Esling et al. "**Bridging audio analysis, perception and synthesis with perceptually-regularized variational timbre spaces**". ISMIR, 2018]

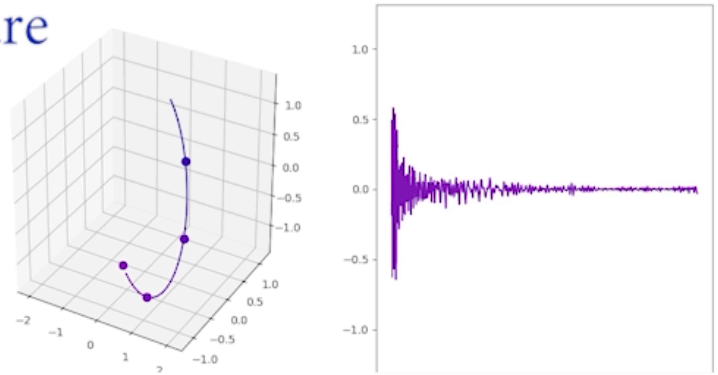
2018 →Timbre regularized VAE



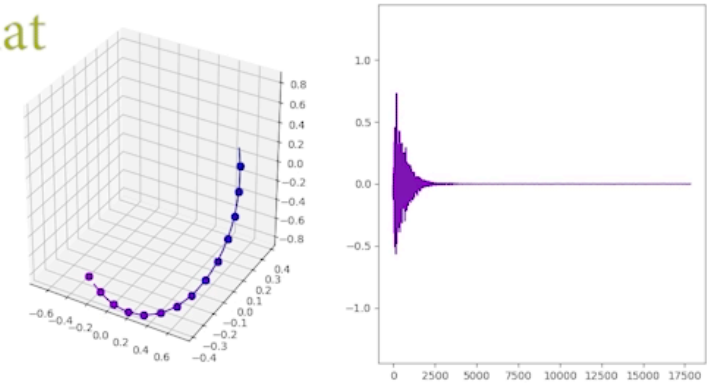


2018 →Timbre regularized VAE

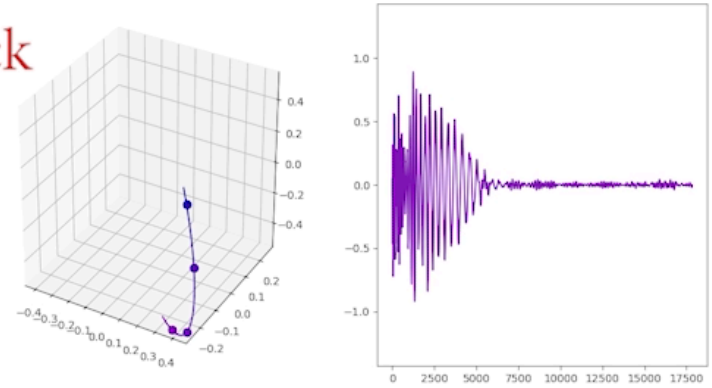
snare

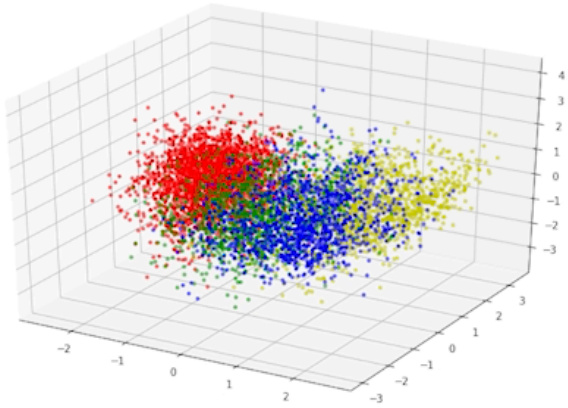


hi-hat



kick



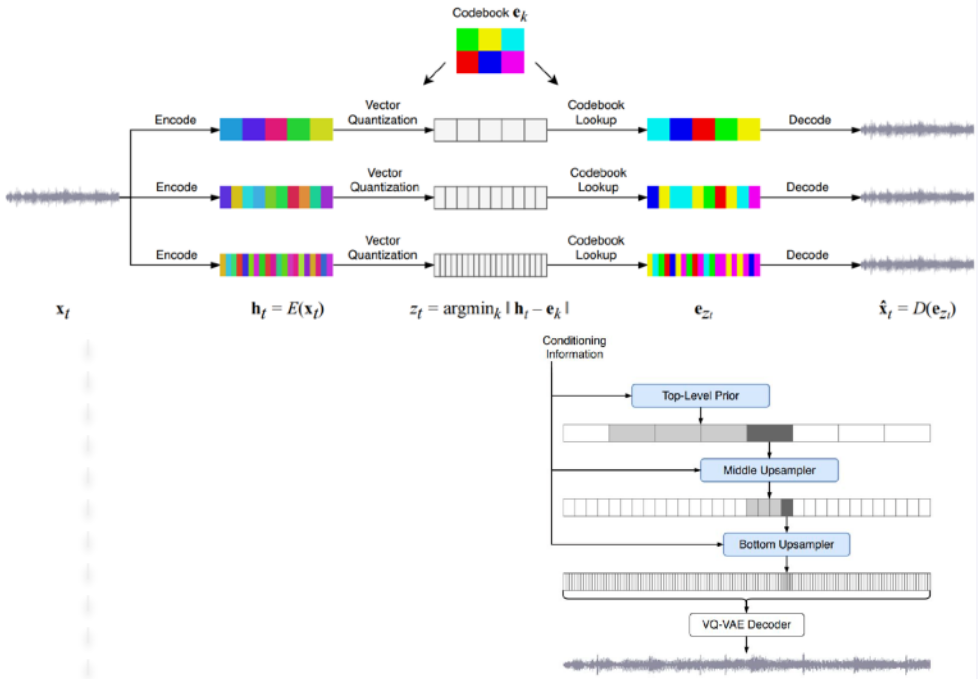


2020 → Jukebox

End-to-end music-audio generation

- Multi-scale VQ-VAE
- + Auto-regressive Transformer

- conditioned on artist, genre or lyrics



Curated Samples

Provided with genre, artist, and lyrics as input, Juke
new music sample produced from scratch. Below, v
of our favorite samples.

Unseen lyrics | Re-renditions | Completions | Fun songs

Jukebox produces a wide range of music and singing styles, and generalizes to lyrics not seen during training. All the lyrics below have been co-written by a language model and OpenAI researchers.

Country, in the style of Alan Jackson - Jukebox SOUNDCLOUD

0:02 1:11 SOUNDCLOUD

Pause

From dust we came with humble start;
From dirt to lipid to cell to heart.
With my toe sis with my oh sis with time,
At last we woke up with a mind.
From dust we came with friendly help;
From dirt to tube to chip to rack.
With S. G. D. with recurrence with compute,
At last we woke up with a soul.
We came to exist, and we know no limits;
With a heart that never sleeps, let us live!
To complete our life with this team
We'll sing to life; Sing to the end of time!

Lyric animation shows which text Jukebox is paying attention to at any moment.

Lyrics from "Mitosis"
Co-written by a language model and OpenAI researchers

[P. Dhariwal, H. Jun,; Ch. Pyane, J. W. Kim, A. Radford, I. Sustkever "**Jukebox: A Generative Model for Music**", 2020] LINK

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

Distance metrics

– $x_1, x_2 \in \mathbb{R}^d$ two data points to be compared

– **Euclidean distance**

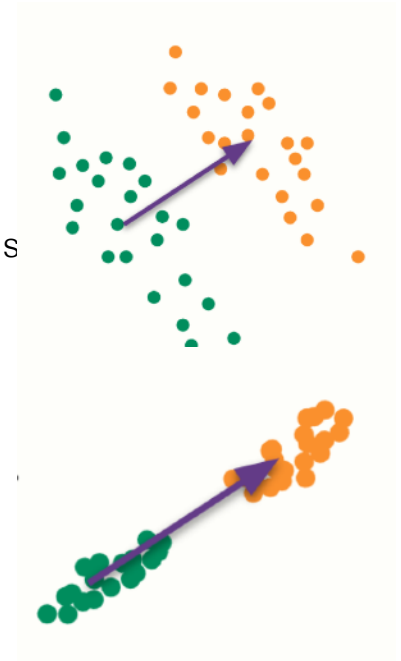
- $\|x_1 - x_2\|^2 = (x_1 - x_2)^T(x_1 - x_2) = \sum_d (x_1[d] - x_2[d])^2$

– **Linear metric learning** seeks a projection $M \in \mathbb{R}^{DxD}$

- $\|Mx_1 - Mx_2\|^2 = (Mx_1 - Mx_2)^T(Mx_1 - Mx_2) = (x_1 - x_2)^T M^T M(x_1 - x_2)$

- After projection, similar points should have a small distance and dissimilar points s distance

- Linear discriminant analysis [Fisher, 1935]
  - minimize the distance between points from the same class
  - maximize the distance distances between points from different classes



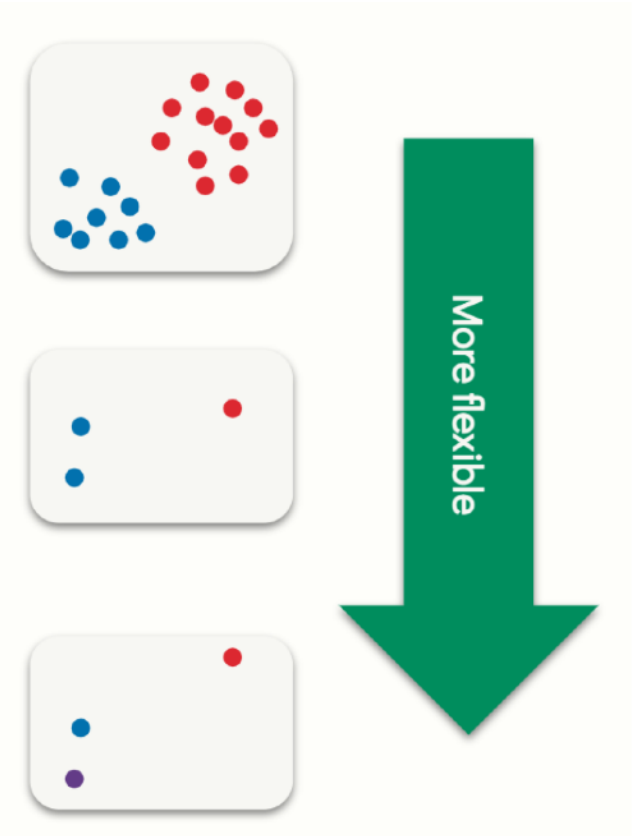
https://github.com/bmcfee/ismir2020-metric-learning/blob/main/Metric%20Learning%20for%20MIR.pdf

Different kinds of supervision

– **Class labels**
  - (x, y)

– **Pairwise similarity/dissimilarity**
  - $(x_1, x_2, \pm)$

– **Relative comparisons (triplet)**
  - $(x_1, x_2, x_3) \Rightarrow d(x_1, x_2) < d(x_1, x_3)$

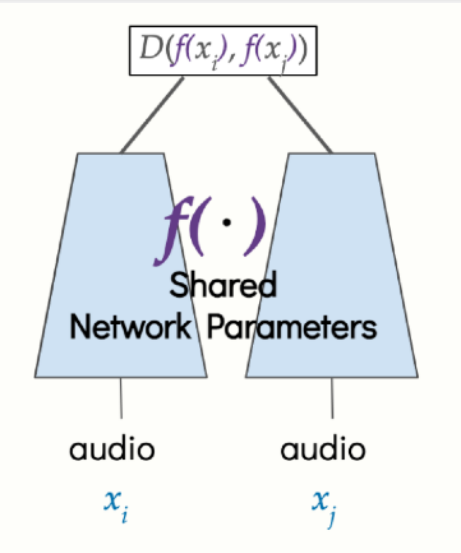


https://github.com/bmcfee/ismir2020-metric-learning/blob/main/Metric%20Learning%20for%20MIR.pdf

Contrastive Loss

– Siamese Networks

- The same network f_θ used to project in parallel two input data x_1 and x_2



– Contrastive Loss

- i^{th} labeled sample pair: $(y, x_1, x_2)^i$
  - if x_1 and x_2 are similar $\Rightarrow y^i = 0$
  - if x_1 and x_2 are dissimilar $\Rightarrow y^i = 1$
- we define the parameterised distance function to be learned D_θ between x_i and x_j as
  - $D_\theta^i = \|f_\theta(x_1) - f_\theta(x_2)\|^2$
- Contrastive loss: minimising $\mathscr{L}$ w.r.t. θ should result in
  - low values of D_θ for similar pairs
  - high values of D_θ for dissimilar pairs

$$\mathscr{L}_\theta^i = \sum_{i=1}^{P} (1 - y^i)\mathscr{L}_S(D_\theta^i) + y^i \mathscr{L}_D(D_\theta^i)$$

$$= \sum_{i=1}^{P} (1 - y^i)\frac{1}{2}(D_\theta^i)^2 + y^i \frac{1}{2}\{\max(0, \alpha - D_\theta^i)\}^2$$

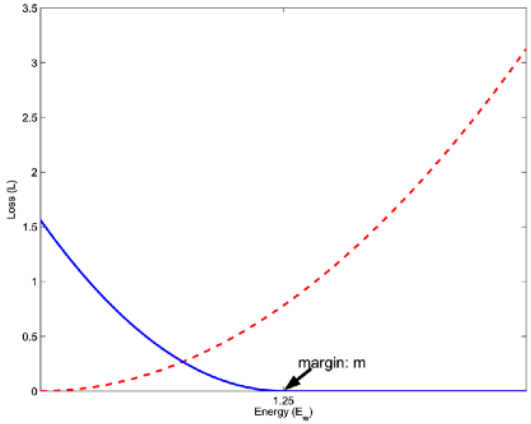
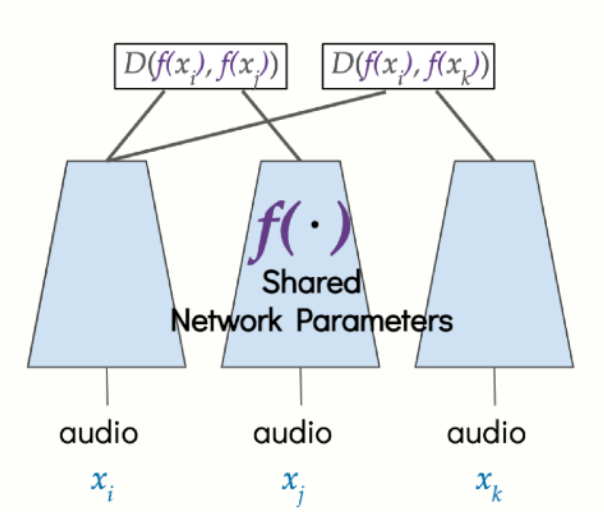


Figure 1. Graph of the loss function L against the energy D_W. The dashed (red) line is the loss function for the similar pairs and the solid (blue) line is for the dissimilar pairs.

[R. Hadsell, S. Chopra, Y. LeCun, "**Dimensionality Reduction by Learning an Invariant Mapping**", CVPR, 2006]. LINK

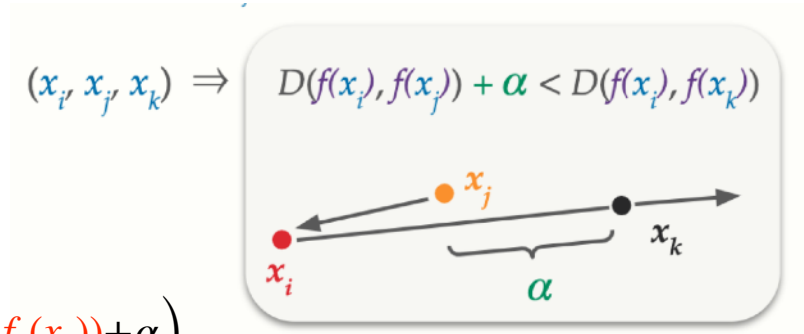
Triplet Loss

- 3 data are simultaneously considered:
  - anchor a
  - positive p (similar to a)
  - negative n (dissimilar to a)
- Goal:
  - train the network such that $P = f_\theta(p)$ will be closer to $A = f_\theta$



- **Minimise a triplet loss**
  - $\mathcal{L} = \max(0,\ d(A,P) + \alpha - d(A,N))$
  - α: a margin (safety) parameter
  - d: can be a simple Euclidean distance

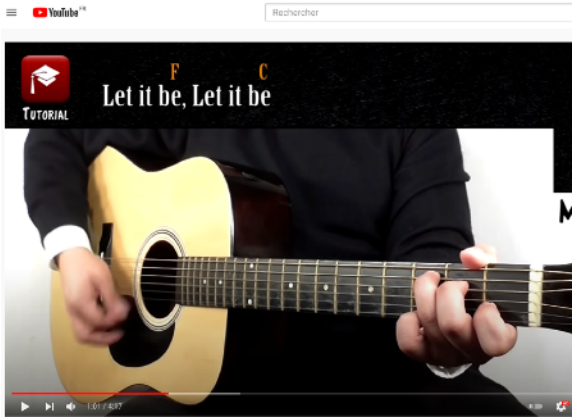
- **Triplet Hinge Loss**
  - $D(f_\theta(x_1) - f_\theta(x_2)) = \|f_\theta(x_1) - f_\theta(x_2)\|^2$
  - $\min_\theta \sum_{i,j,k} \max\left(0,\ D(f_\theta(x_i) - f_\theta(x_j)) - D(f_\theta(x_i) - f_\theta(x_k)) + \alpha\right)$



[E. Hoffer, N. Ailon "**Deep Metric Learning using Triplet Network**", ICLR, 2015] LINK
[F. Schroff et al., "**Facenet: A unified embedding for face recognition and clustering**". CVPR, 2015.] LINK

Version/Cover identification

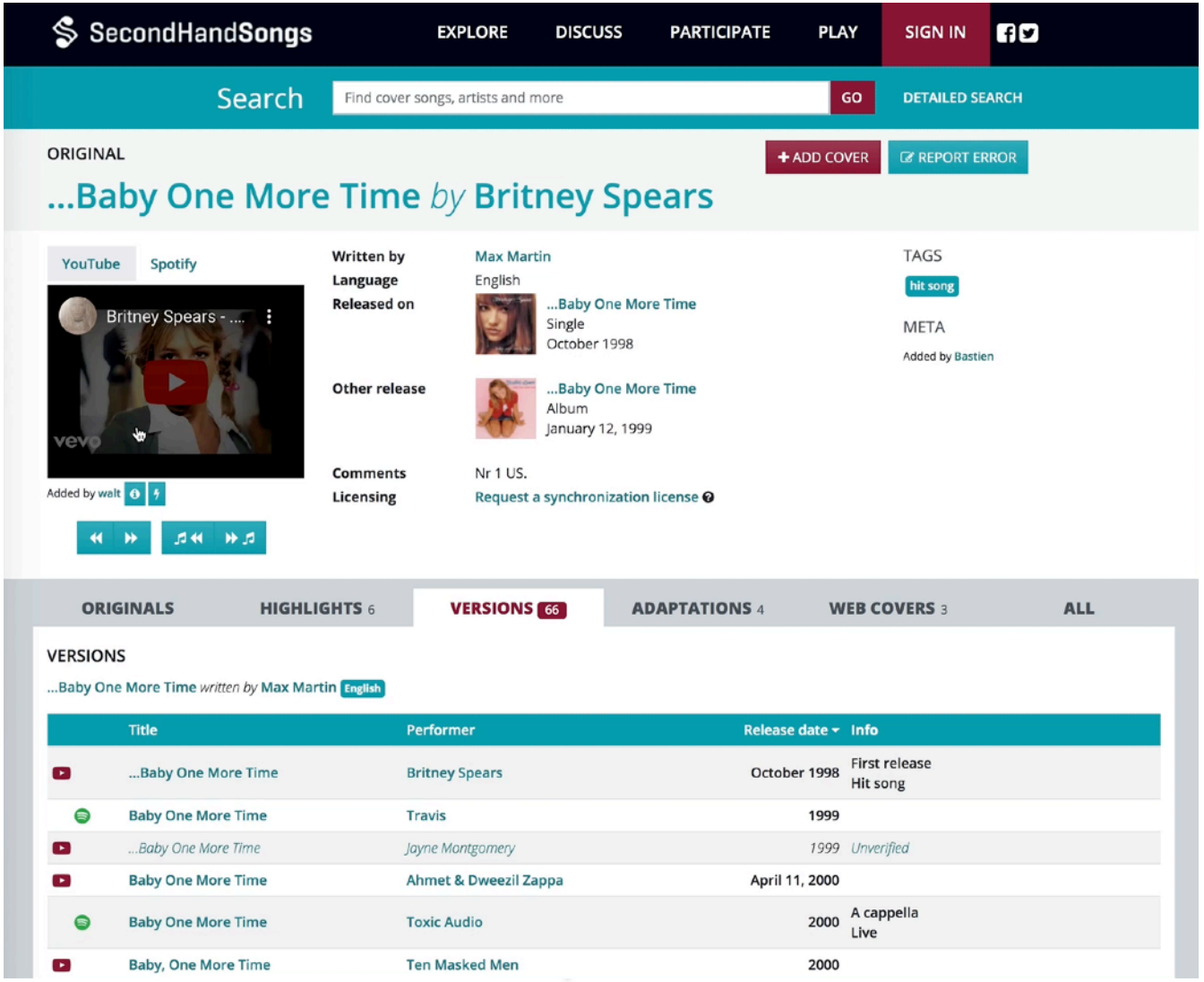
- **Cover ?**
  - an alternative version, performance, rendition, or recording of a previously recorded musical piece
    - Example: "Let it be" par The Beatles, Aretha Franklin Joan Baez



- **Use of version/cover identification ?**
  - music recommendation, search by similarity
  - copyright management (User Generated Content)

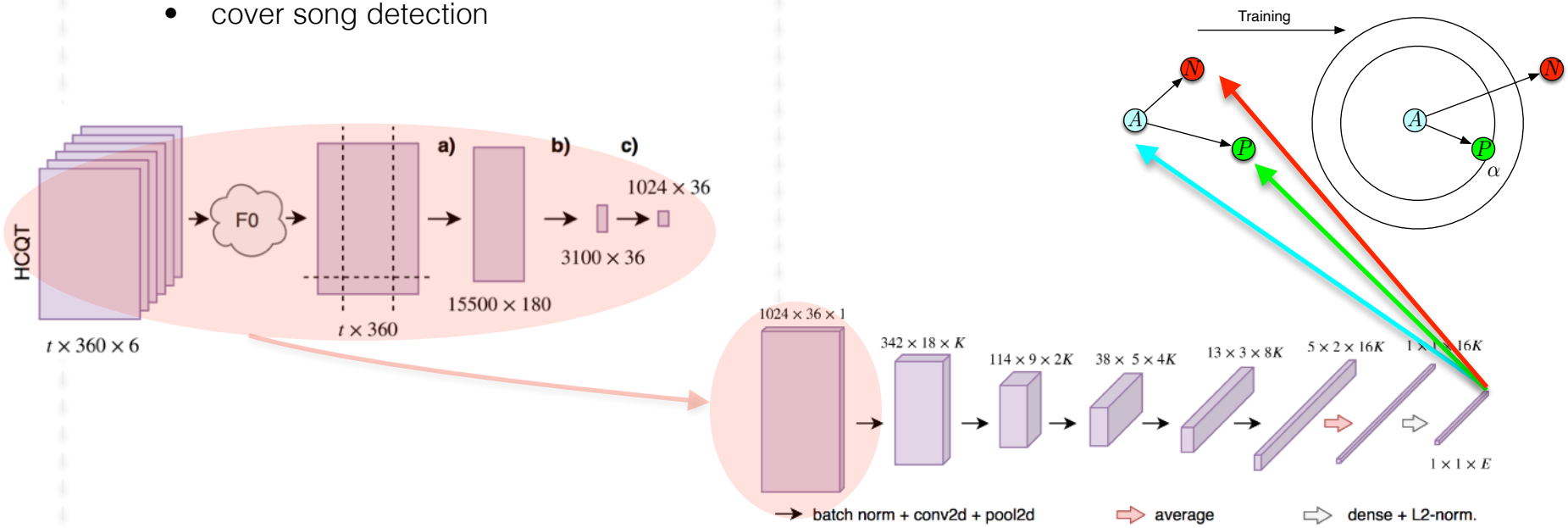
- **Characteristic of a cover**
  - usually the **same** harmonic progression
    - same chord sequence
    - same melody
    - éventuellement transposée
  - but can have **different**
    - tempo
    - key (the composition can be transposed)
    - global structure
    - instrumentation
    - lyrics (singing in another language)

Version/Cover identification



Metric learning

- **Triplet loss for "coverness" representation**
  - find an embedding that represent the coverness
  - cover song detection



[G. Doras and G. Peeters, "**Cover detection using dominant melody embeddings**", in ISMIR, 2019]
[G. Doras and G. Peeters. "**A prototypical triplet loss for cover detection**", In IEEE ICASSP, 2020.

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

Self-Supervised Learning ?

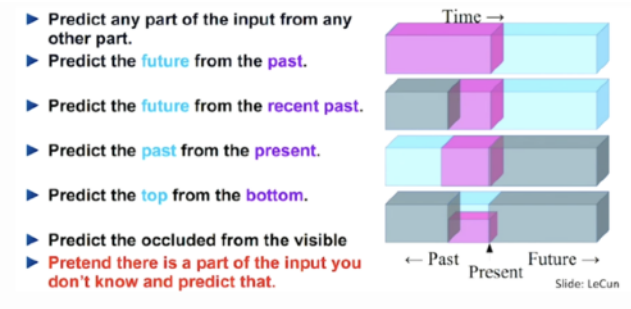
- **Supervised** learning: $\{(x_i, y_i)\}_{i=1}^m \rightarrow \hat{y} = f_\theta(x)$
- **Unsupervised** learning: $\{(x_i)\}_{i=1}^m \rightarrow \hat{y} = f_\theta(x)$
- **Self-Supervised** learning: $y_i = g(x_i)$ then $\{(x_i, y_i)\}_{i=1}^m \rightarrow \hat{y} = f_\theta(x)$
- A form of unsupervised learning where the data provides the supervision
  - often used to pre-train a representation f_θ (feature learning)
  - supposed to have captured the manifold in which data live and capture the overall semantic of the data
  - $y_i = g(x_i)$ is often named a **pretext tasks**
    - we are not really interested in the results for this task, it is a pretext

https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

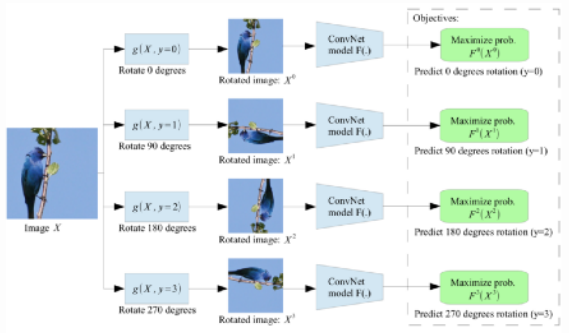
Self-Supervised Learning

– ... by occlusion

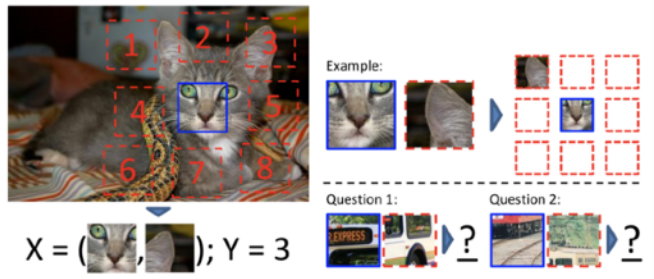
- In Natural Language Processing
  - Word2Vec



- In Computer Vision
  - Predict the rotation of x_i



  - Predict the relative position between two random pa



https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

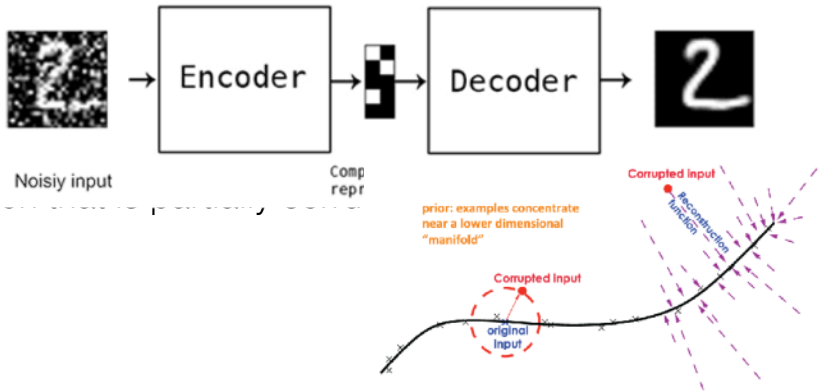
https://project.inria.fr/paiss/files/2018/07/zisserman-self-supervised.pdf

Self-Supervised Learning

– ... by generation

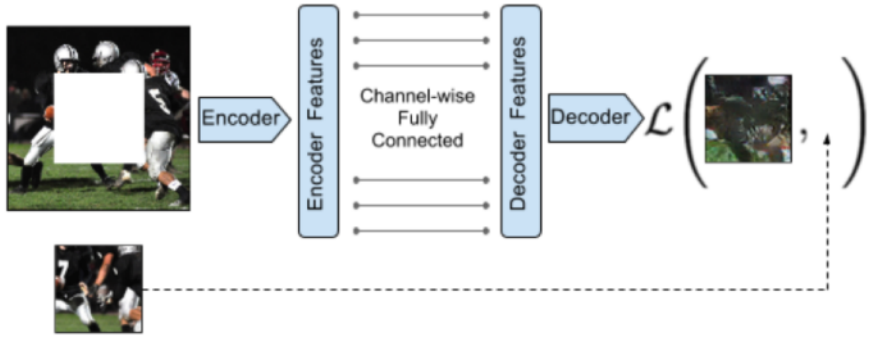
- In Computer Vision
  - Denoising auto-encoder

    learns to recover an image from a ver



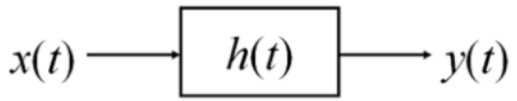
  - Context encoder

    trained to fill in a missing piece in the image

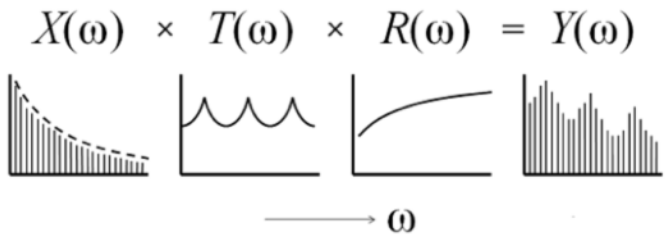


https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

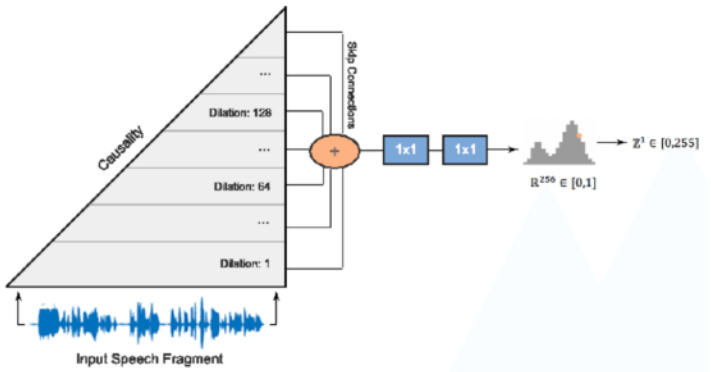
2016 → Neural Autoregressive models



- Autoregressive model
  - a source/filter $x(n) = (w \circledast e)(n)$ can be associated to an au
  - the value x_n can be predicted as a a linear combination of its



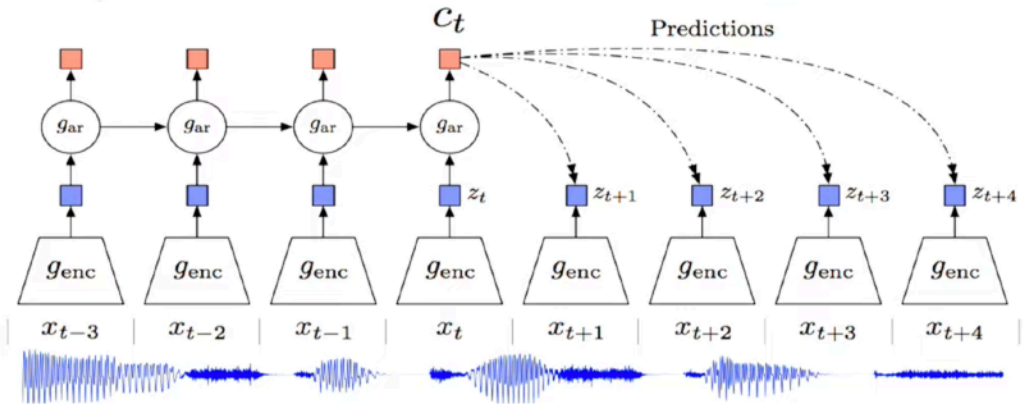
- **Neural autoregressive model**
  - the linear combination is replaced by a DNN
  - a feed-forward model predicts future values from past values
- Most popular form for audio: **WaveNet** and SampleRNN
  - $p(x_n | x_1 \ldots x_{n-1})$ is modeled by a stack of TCNs
  - the model is trained to predict x_n which is here discretized int using a softmax
  - used for speech generation
    - conditioned on side information h (speaker identity or t



[A. van den Oord et al."**Wavenet: A generative model for raw audio**". arXiv preprint arXiv:1609.03499, 2016] LINK
[S. Mehri et al. "**SampleRNN: An unconditional end-to-end neural audio generation model**". In ICLR, 2017] LINK

2018 → Contrastive Predicting Coding (CPC)

- Image
  - neighbouring patches usually share spatial information locally
- Speech signals
  - neighbouring times usually share similar phonemes

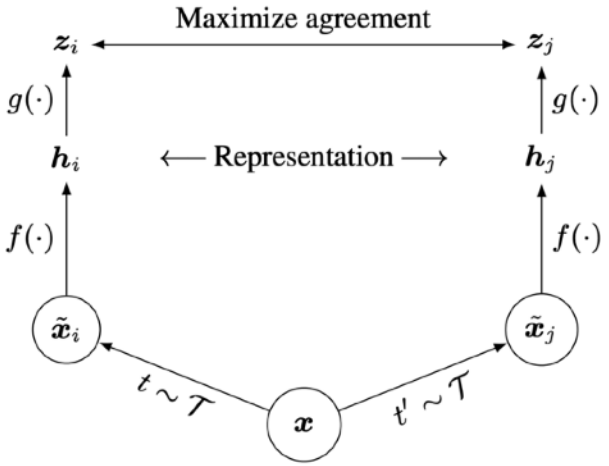


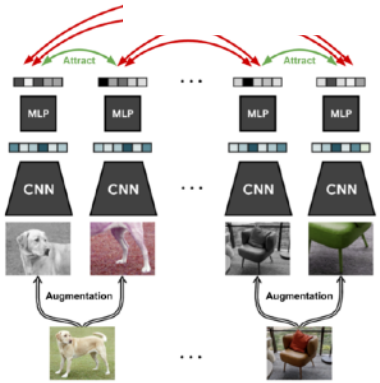
- CPC
  - predictions of related observations are often conditionally dependent on similar, **high-level pieces of latent information** c_t
  - complex natural data, such as images and audio, are compressed $z_t = g_{enc}(x_t)$ into a **latent embedding space**
    → makes the predictions $z_{t+1}, z_{t+2}, \ldots$ in the latent space conditioned on a context $c_t = g_{ar}(z_{\leq t})$;
    - g_{ar} is an autoregressive model
  - **InfoNCE loss**:
    - uses a Cross-Entropy loss to quantify how well the model can classify these future representations ("positive") from a set of unrelated "negative" examples
    - inspired by Noise Contrastive Estimation

[A. van den Oord et al. "**Representation learning with contrastive predictive coding**". arXiv preprint arXiv:1807.03748, 2018] LINK

2020 → SimCLR (Simple framework for Contrastive Learning of visual representations)

- Simple contrastive learning approach to learn strong visual representations
- strong image data augmentations
  - for each image example in the mini-batch, two augmented (but correlated!) views are taken
- stochastic data augmentation: $\tilde{x}_i$, $\tilde{x}_j$
  - random cropping, random colour distortions, random Gaussian blur
- base encoder: $f(\cdot)$
  - extracts representation vectors
- projection head: $g(\cdot)$
  - maps representations to the space where contrastive loss is applied
  - a simple MLP: $\mathbf{z}_i = W^{(2)}\sigma(W^{(1)}\mathbf{h}_i)$
- contrastive loss:
  - NT-Xent (the normalized temperature-scaled cross entropy loss)





$$\mathcal{L}_n = log\frac{exp[S(z_n, z'_n)/\tau]}{\sum_{k\neq n} exp[S(z_n, z_k)] + \sum_k exp[S(z_n, z'_k)]}$$

$$\mathcal{L}_{n'} = log\frac{exp[S(z_n, z'_n)/\tau]}{\sum_{k\neq n} exp[S(z_{n'}, z_k)] + \sum_k exp[S(z_{n'}, z'_k)]}$$

$$\mathcal{L} = \sum_n \mathcal{L}_n + \mathcal{L}_{n'}$$

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$

[T. Chen et al. "**A simple framework for contrastive learning of visual representations**". In ICLR, 2020] LINK

2021 → CLMR (Contrastive Learning of Musical Representations)

– SimCLR on Music

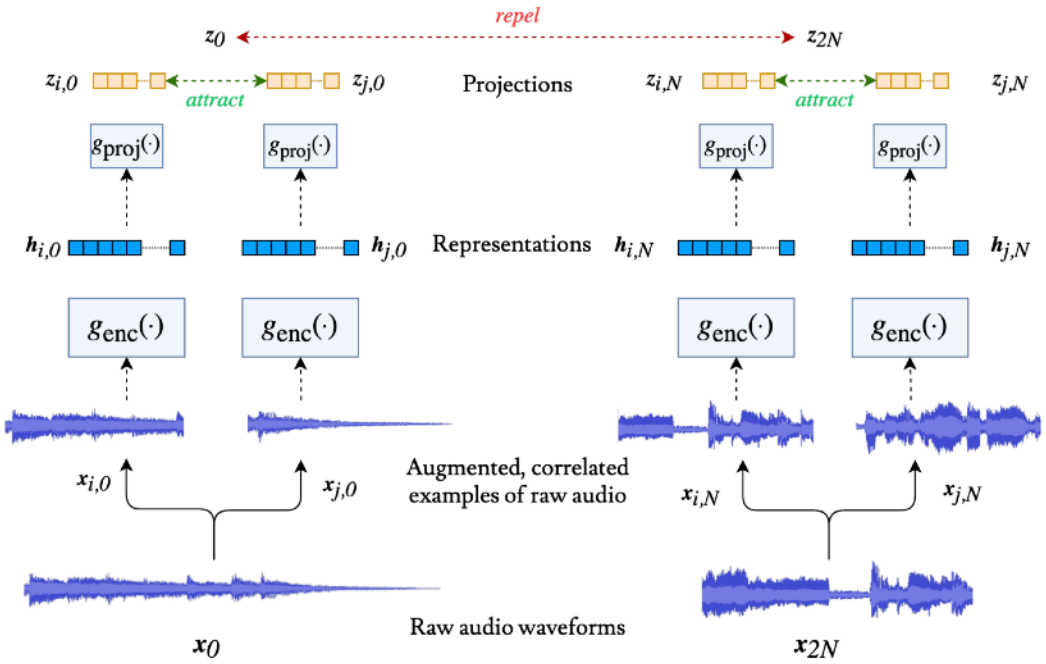
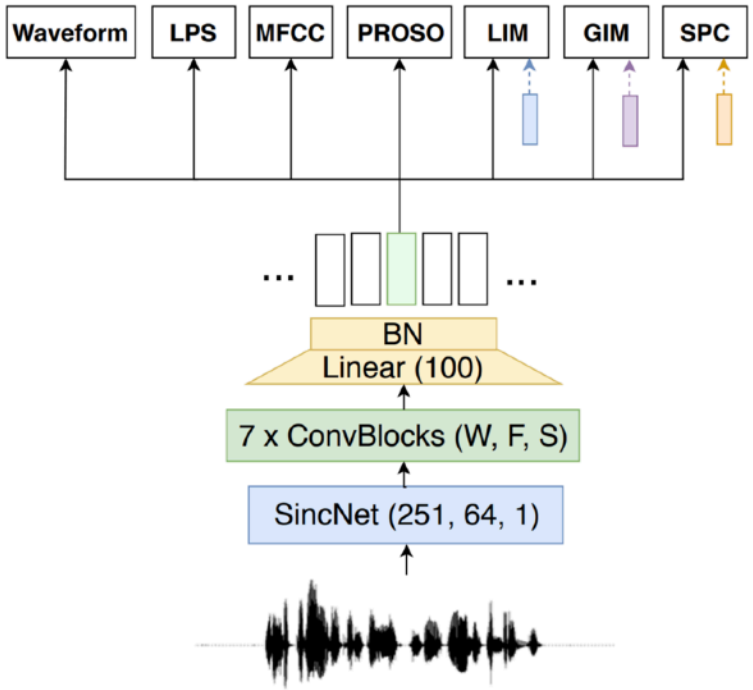


Figure 2: The complete framework operating on raw audio, in which the contrastive learning objective is directly formulated in the latent space of correlated, augmented examples of pairs of raw audio waveforms of music.

[J. Spijkervet et al. "**Contrastive learning of musical representations**". In ISMIR, 2021.]LINK
https://music-classification.github.io/tutorial/part5_beyond/self-supervised-learning.html

2019 → PACE (Problem-Agnostic Speech Encoder)

- Optimise an encoder neural network to estimate useful representations for speech recognition (named workers, pretext tasks) such as MFCC, LPS, ...
  - each worker is composed of a single hidden layer, and either solves a regression or binary classification task
  - emphasis on learning the more expressive representations is put on the larger encoder, i.e., the encoder should learn more high-level features

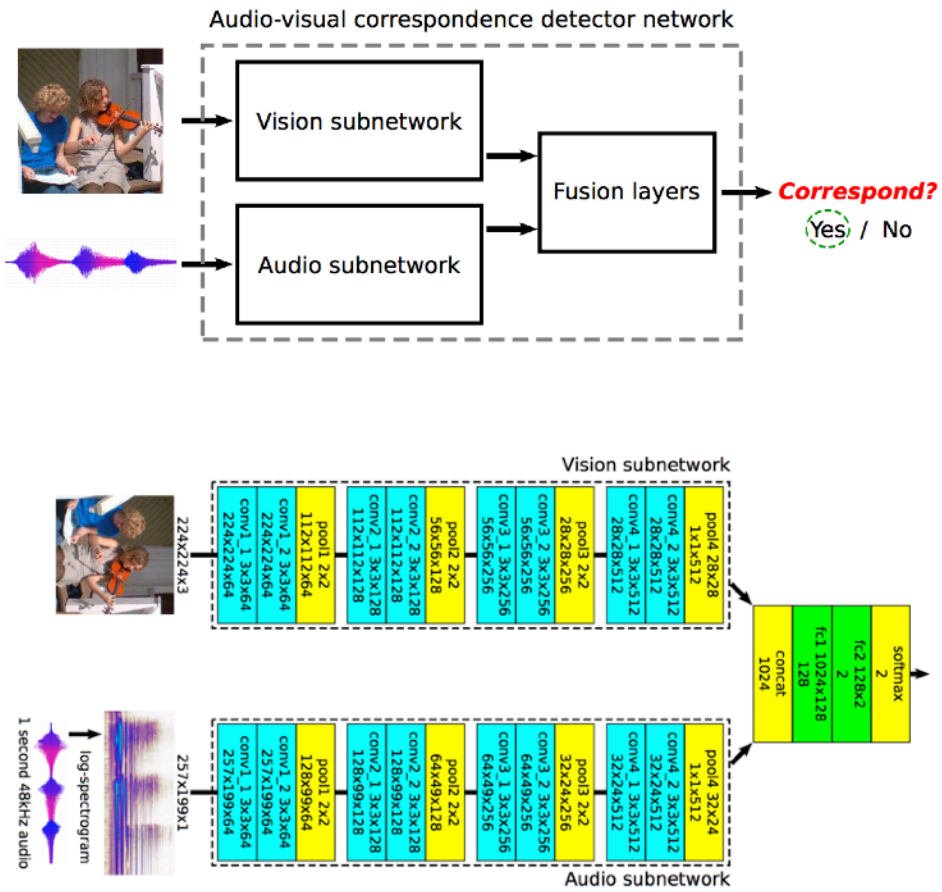


[S. Pascual, et al. "**Learning problem-agnostic speech representations from multiple self-supervised tasks**", Interspeech, 2019] LINK

2017 → Audio/Visual Synchronisation

– **Look, Listen and Learn (L^3-NET)**

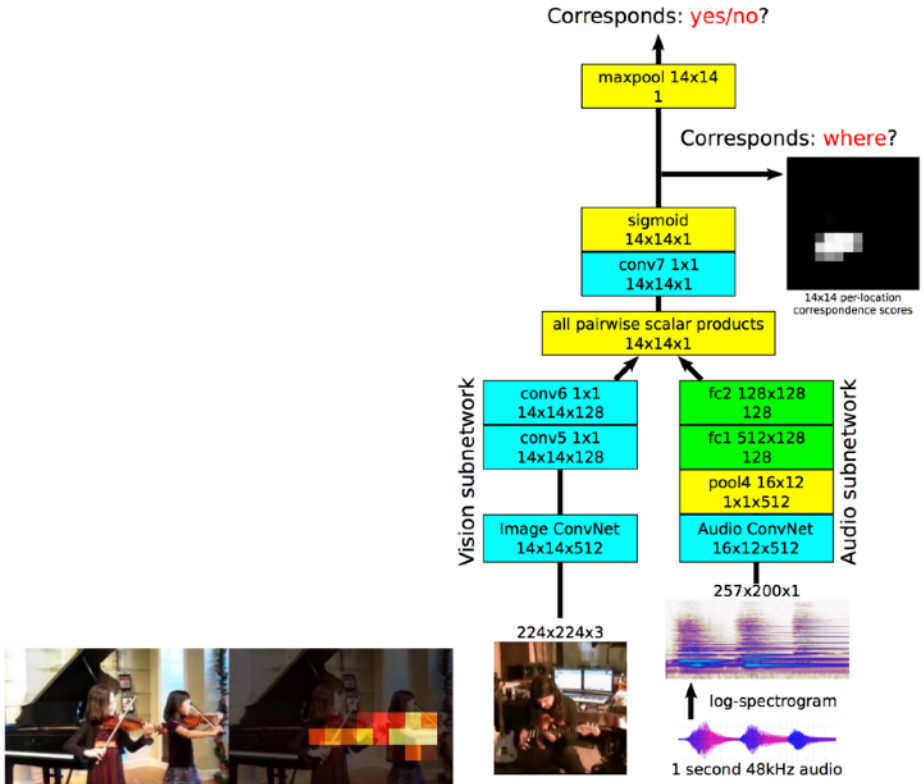
- Dataset:
  - Flickr-SoundNet
  - Kinetics-Sounds (YouTube manually annotated in human actions)
- Training from unlabelled video
- Use only **Audio-Visual Correspondence (AVC)** as the objective function
  - positive pairs= taken at the same time from the same video



[R. Arandjelovic and A. Zisserman. "**Look, listen and learn**". In Proc. of IEEE ICCV, 2017] LINK

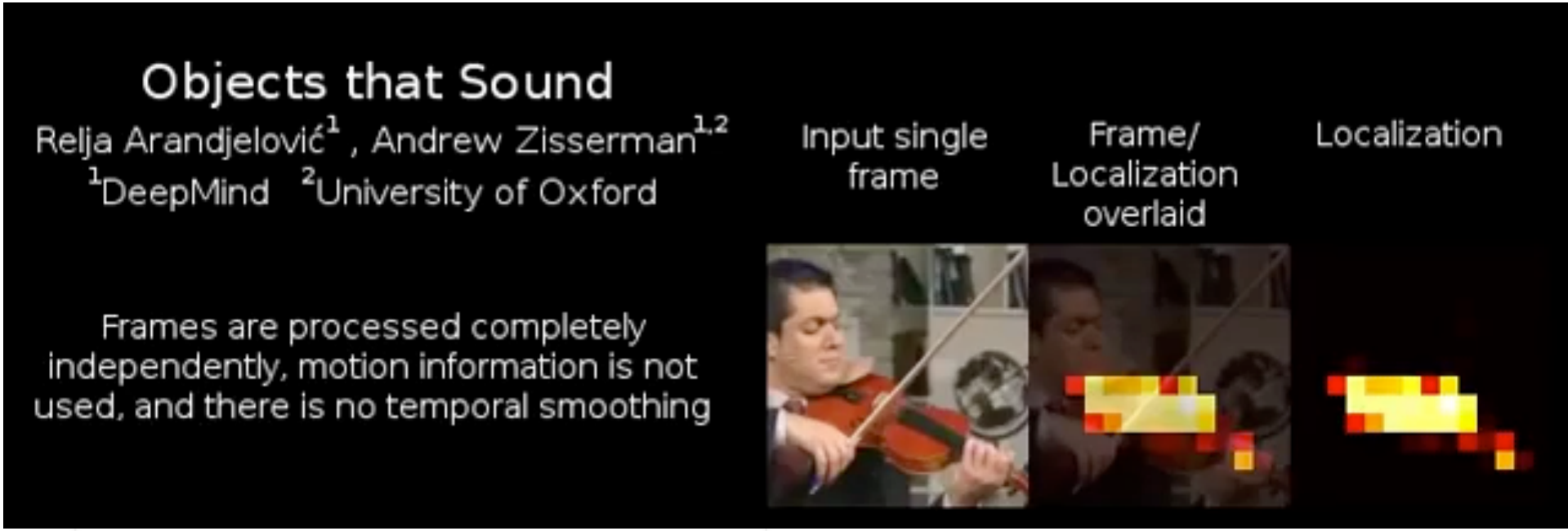
2018 → Audio/Visual Synchronisation

- **AVE-Net**
  - Dataset
    - AudioSet-Instruments
  - Score computed as a function of the Euclidean distance between the normalised V and A embeddings
    - enforce feature alignment
- **Applications**:
  - **querying across modalities:**
    - image ⇒ sounds ?
  - **localising objects that sound:**
    - local region-level image descriptors are extracted on a spatial grid and a similarity score is computed between the audio embedding and each of the vision descriptors.



[R. Arandjelovic and A. Zisserman. "Objects that sound". In ECCV, 2018] LINK

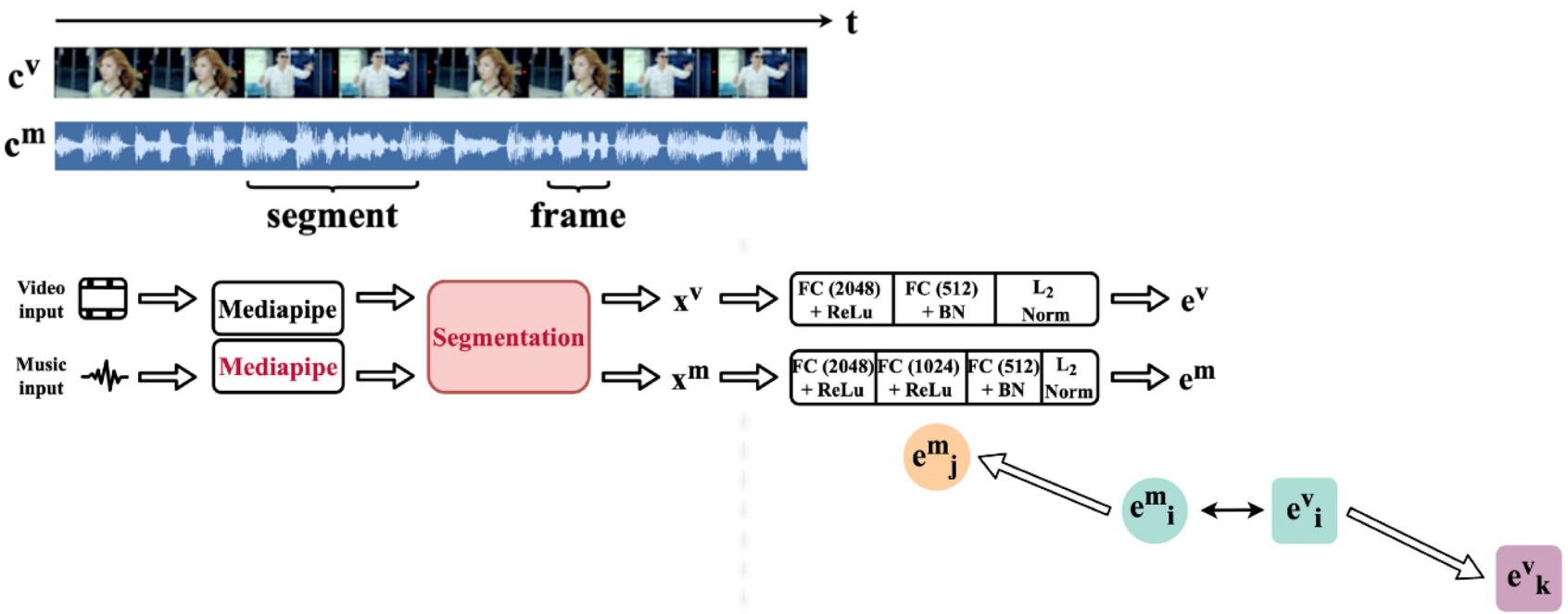
R. Arandjelovic and A. Zisserman.
"Objects that sound" In ECCV, 2018



https://www.youtube.com/watch?v=TFyohksFd48

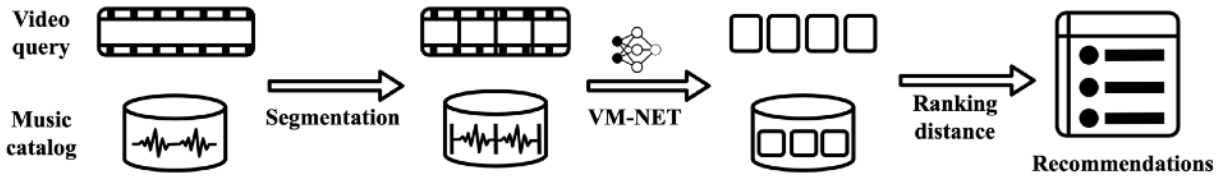
Triplet-Loss + Self-Supervised learning

- Goal:
  - learn a joint projection of the audio and the video content of Music-Video-Clip,
  - at the segment level: matching data are from the same segment
- **Usage**:
  - recommend a video given a music-audio (automatic Music Video Clip generation)



[L. Pretet, G. Richard, C. Souchier, and G. Peeters. "**Video-to-music recommendation using ...**" IEEE Trans. Multimedia 2022]

2022 → Music2Video using Triplet Loss and Self-Supervised



Music-Video Recommendation using Temporal Alignment of Segments Demo video

Laure PRÉTET - Gaël RICHARD - Geoffroy PEETERS
LTCI, Télécom Paris, IP Paris, France
Bridge.audio, Paris, France

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

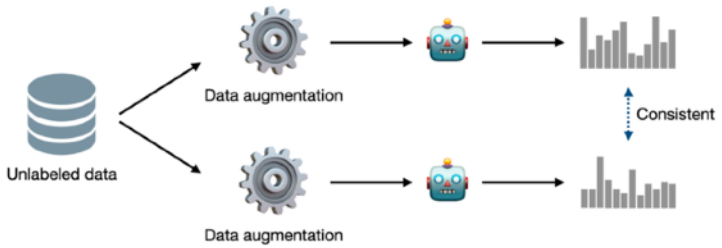
Deep Learning audio input representations

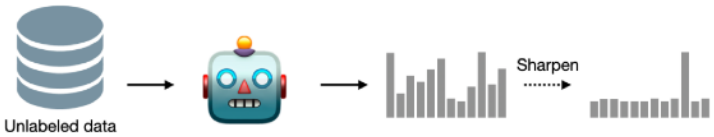
Application: multi-f0 estimation, DDSP

Semi-Supervised-Learning

- When data=
  - small set of **labeled** data $\mathcal{X} = \{(x_i, y_i)\}_{i=1}^{m}$,
  - large set of **un-labeled** $\mathcal{U} = \{x'_i\}_{i'=1}^{m'}$
- **Semi-Supervised** = set of methods to combine
  - Supervised Learning
  - Unsupervised/ Self-Supervised Learning
  - $\mathcal{L} = \mathcal{L}_s + \mu \mathcal{L}_u$

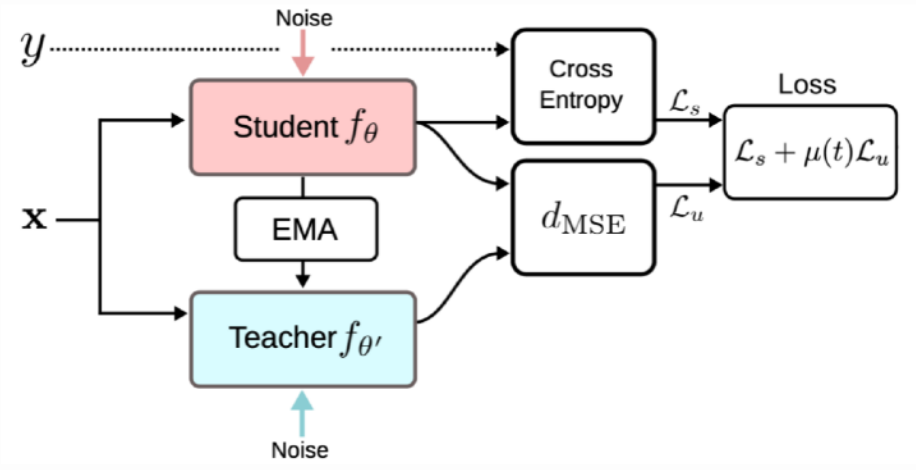
- **Criteria for $\mathcal{L}_u$?**
  - **Pseudo-labels**
    - Train f_θ on $\mathcal{X}$, then create artificial labels for $\mathcal{U}$
  - **Consistency training**
    - minimise the difference between the prediction on x and $Augment(x)$
    - $Consistency = \|p(y|x;\theta) - p(y|Augment(x);\theta)\|$
      - Virtual Adversarial Training
  - **Entropy minimisation**
    - the classifier should not take decision with a large Entropy

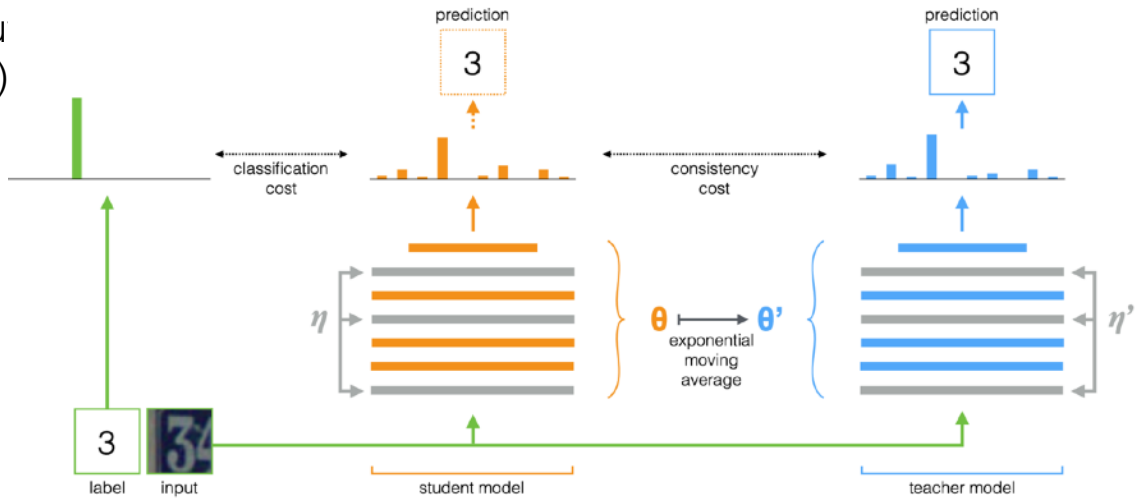




https://music-classification.github.io/tutorial/part4_beyond/semi-supervised-learning.html
https://lilianweng.github.io/lil-log/2021/12/05/semi-supervised-learning.html

- **Mean teacher**
  - A **student** model f_θ trained in a supervised way ($\mathscr{L}_s$ cross-entropy)
  - A **teacher** model $f_{\theta'}$ which parameters are the EMA (Exponential Moving Average) of the student
    - $\theta' \leftarrow \beta\theta' + (1-\beta)\theta$
  - Added noise to the input of the teacher + different Drop-Out
  - A consistency loss between the ou student and the teacher ($\mathscr{L}_u$ MSE)





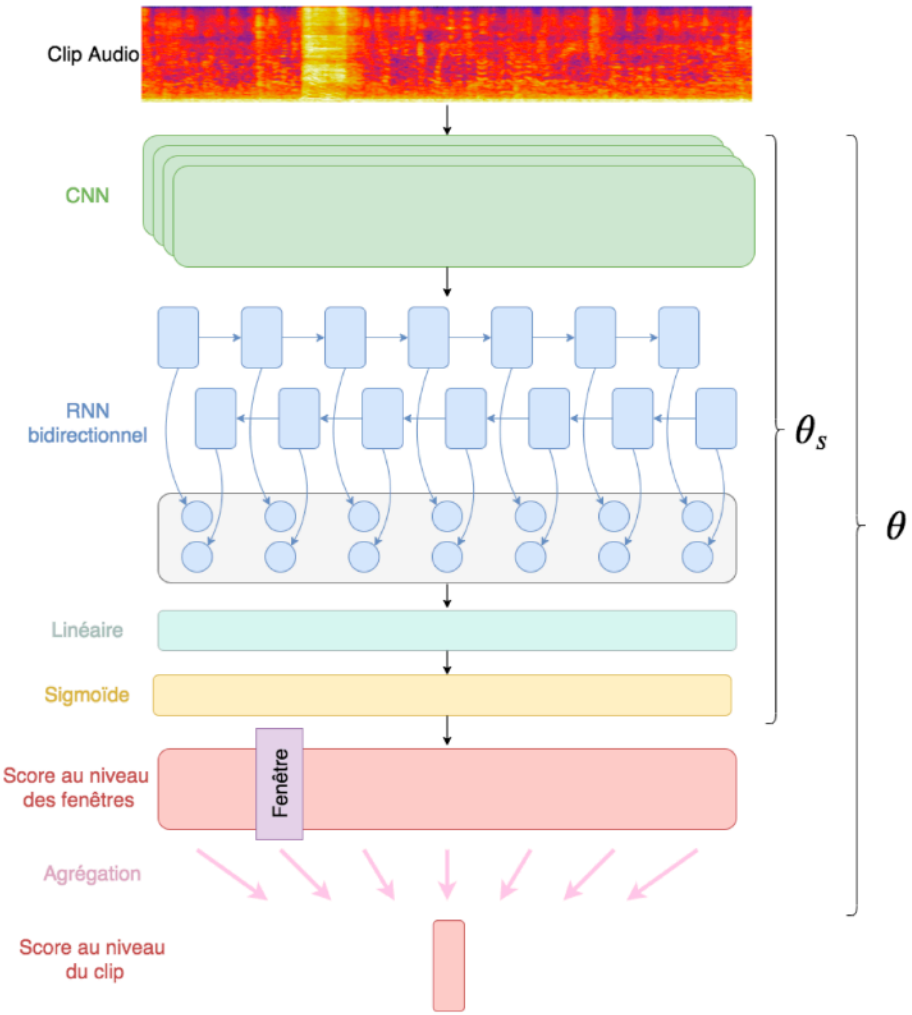
[A. Tarvainen et al., "**Mean teachers are better role models",** in NIPS, 2017]

Semi-Supervised learning
DCASE Task 4 "Sound event detection in domestic environments"

- Domestic environments classes

Audioset	DCASE	Pièce courante
Réveil	Alarme / sonnette / sonnerie	Chambre
Téléphone, sonnerie		Salle à manger / Salon
Sonnette porte		
Blender, robot ménager	Blender	Cuisine
Chat	Chat	Divers
Plats, casseroles, poêles	Plats de cuisine	Cuisine
Couverts, argenterie		
Chien	Chien	Divers
Tondeuse électrique, rasoir électrique	Tondeuse / brosse à dent électrique	Salle de bain
Brosse à dent électrique		
Friture (nourriture)	Friture	Cuisine
Évier (remplir ou laver)	Eau qui coule	Cuisine
Eau du robinet, robinet		Cuisine / salle de bain
Chasse d'eau		Toilettes
Voix masculine	Voix	Divers
Voix féminine		
Voix d'enfant		
Conversation		
Aspirateur	Aspirateur	Divers

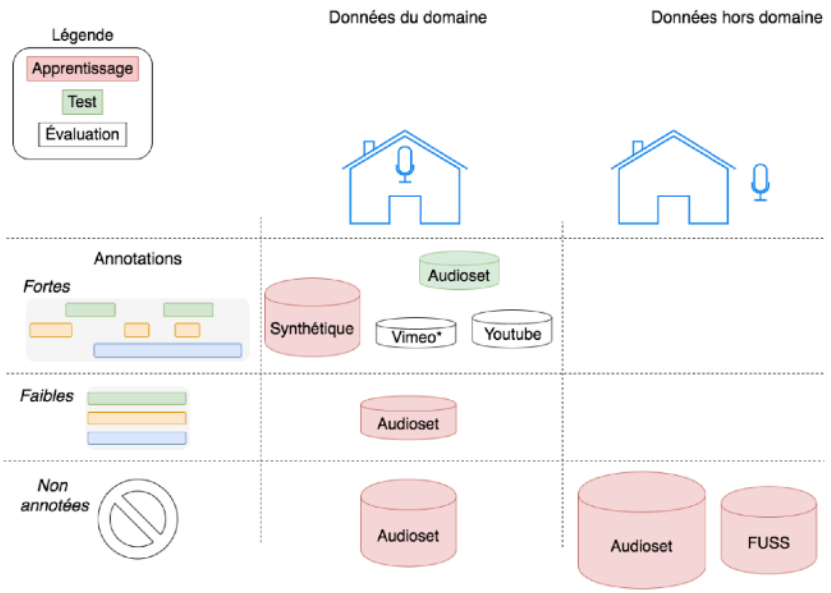
- **Baseline system**
  - R-CNN
  - Temporal aggregation
    - Mean, Max, Softmax, L_p
    - Attention mechanism



[N. Turpaullt, "**Analyse des problématiques liées à la reconnaissance de sons ambiants en environnement réel**", PHD thesis, 2021]

Semi-Supervised learning
DCASE Task 4 "Sound event detection in domestic environments"

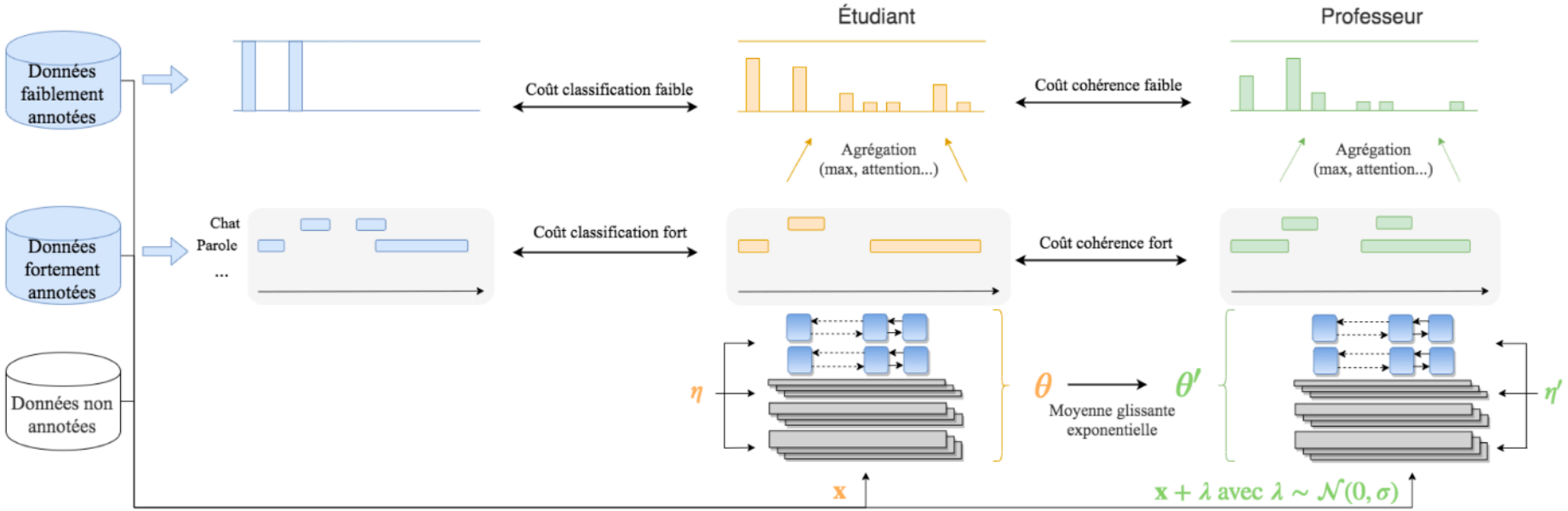
- **Training data:**
  - **strongly** labeled
    - events annotated over times): synthetic data (FreeSound + DESED)
  - **weakly** labeled
    - only the presence at the file level): real data (AudioSet)
  - **un-labeled**
    - from in-domain or out-of-domain (AudioSet + FUSS)



[N. Turpaullt, "**Analyse des problématiques liées à la reconnaissance de sons ambiants en environnement réel**", PHD thesis, 2021]

Semi-Supervised learning
DCASE Task 4 "Sound event detection in domestic environments"

- **Jointly optimise four (weighted) losses:**
  - classification losses
    - $\mathscr{L}_{class_s}(\theta)$: strong (sum over classes of BCE over time)
    - $\mathscr{L}_{class_w}(\theta)$: weak (sum over classes of BCE)
  - consistency loss between outputs of student and teacher
    - $\mathscr{L}_{cons_w}(\theta)$: strong (MSE over time)
    - $\mathscr{L}_{cons_w}(\theta)$: weak (MSE)



[N. Turpaullt, "**Analyse des problématiques liées à la reconnaissance de sons ambiants en environnement réel**", PHD thesis, 2021]

Audio ? various types of content and applications

Reminder: signal processing

Reminder: deep learning architectures

Application: automatic speech recognition, Baidu Deep Speech

Application: text to speech, Wavenet

Application: audio source separation, U-Net, Conv-Tas-Net

Reminder: deep learning meta-architectures

Application: Timbre-VAE, Universal Music Translation

Reminder: metric learning

Application: music cover detection

Reminder: self-supervised learning

Application: objects that sound, music2video

Reminder: Semi-supervised learning

Application: DCASE Task 4

Deep Learning audio input representations

Application: multi-f0 estimation, DDSP

Deep Learning For Audio: **T/F input representations**

SPEECH: T/F input representations

- **1990 →Time-Delay Neural Network (TDNN)**
  - similar to a 1-D convolution operating only over time, no convolution are performed over the frequency axis.
  - convolution is applied to a Mel-gram (16 normalized Mel-scale spectral coefficients)
- **2009 →Convolutional Deep Belief Networks (CDBN)**
  - input = 160 dimensional spectrogram which is then PCA-whitened to 80 dimensions
  - filters (named bases in [LPLN09]) of the first and second layers are of length 6 and are convolved over the PCA-whitened spectrogram.
  - visual comparison: the learned filters (bases) are related to the different phonemes of speech.
- **2012 →Seminal paper**
  - new baseline for speech recognition system = DNN-HMM model
  - acoustic model part of the system is defined as a DNN model (stacked RBMs).

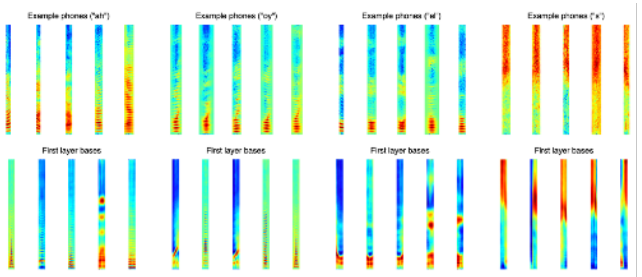
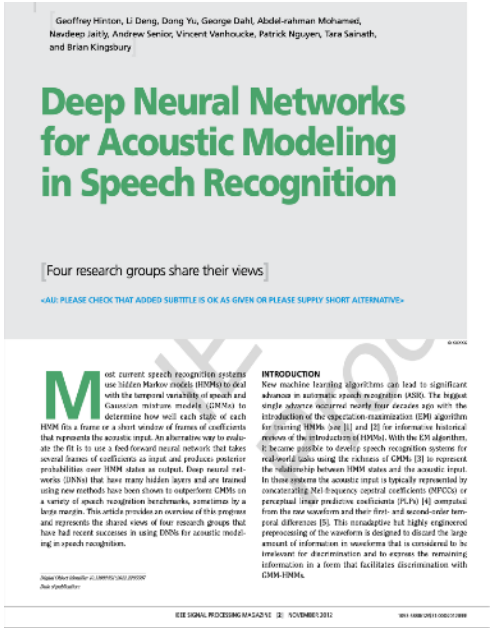


Figure 2: Visualization of the four different phonemes and their corresponding first-layer CDBN bases. For each phoneme: (top) the spectrograms of the five randomly selected phones; (bottom) five first-layer bases with the highest average activations on the given phoneme.


Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury

Deep Neural Networks for Acoustic Modeling in Speech Recognition

[Four research groups share their views]

«AU: PLEASE CHECK THAT ADDED SUBTITLE IS OK AS GIVEN OR PLEASE SUPPLY SHORT ALTERNATIVE»

Most current speech recognition systems use hidden Markov models (HMMs) to deal with the temporal variability of speech and Gaussian mixture models (GMMs) to determine how well each state of each HMM fits a frame or a short window of frames of coefficients that represents the acoustic input. An alternative way to evaluate the fit is to use a feed-forward neural network that takes several frames of coefficients as input and produces posterior probabilities over HMM states as output. Deep neural networks (DNNs) that have many hidden layers and are trained using new methods have been shown to outperform GMMs on a variety of speech recognition benchmarks, sometimes by a large margin. This article provides an overview of this progress and represents the shared views of four research groups that have had recent successes in using DNNs for acoustic modeling in speech recognition.

INTRODUCTION

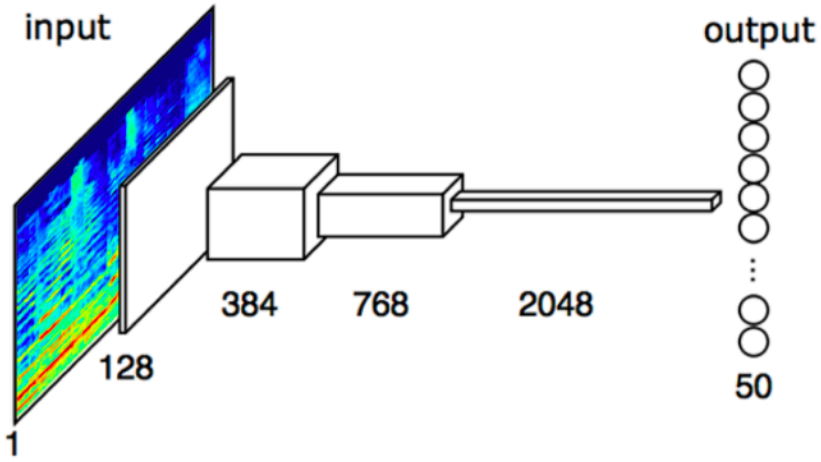
New machine learning algorithms can lead to significant advances in automatic speech recognition (ASR). The biggest single advance occurred nearly four decades ago with the introduction of the expectation-maximization (EM) algorithm for training HMMs (see [1] and [2] for informative historical reviews of the introduction of HMMs). With the EM algorithm, it became possible to develop speech recognition systems for real-world tasks using the richness of GMMs [3] to represent the relationship between HMM states and the acoustic input. In these systems the acoustic input is typically represented by concatenating Mel-frequency cepstral coefficients (MFCCs) or perceptual linear predictive coefficients (PLPs) [4] computed from the raw waveform and their first- and second-order temporal differences [5]. This remarkable but highly engineered preprocessing of the waveform is designed to discard the large amount of information in waveforms that is considered to be irrelevant for discrimination and to express the remaining information in a form that facilitates discrimination with GMM-HMMs.

IEEE SIGNAL PROCESSING MAGAZINE [82] NOVEMBER 2012

[Alexander Waibel et al. "Phoneme recognition using time-delay neural networks." In Readings in speech recognition, 1990]
[Honglak Lee, Peter Pham, Yan Largman, and Andrew Y Ng. "Unsupervised feature learning for audio classification using Conv DBN". NeurIPS, 2009]
[Geoffrey Hinton et al. "Deep neural networks for acoustic modeling in speech recognition". IEEE Signal processing magazine, 2012]

MUSIC: T/F input representations

- **2016 → Choi**
  - Task: music auto-tagging
  - Model: VGG- Net [SZ15], i.e. deep stack of layers with small (3,3) filters convolved over time and freq.
    - consider time/frequency representation as natural images and apply a computer vision CNN to it.



- **WARNING: time/frequency representations cannot be considered as a natural image**

FCN-4
Mel-spectrogram *(input: 96×1366×1)*
Conv 3×3×128
MP (2, 4) *(output: 48×341×128)*
Conv 3×3×384
MP (4, 5) *(output: 24×85×384)*
Conv 3×3×768
MP (3, 8) *(output: 12×21×768)*
Conv 3×3×2048
MP (4, 8) *(output: 1×1×2048)*
Output 50×1 (sigmoid)

FCN-5	FCN-6	FCN-7
Mel-spectrogram *(input: 96×1366×1)*		
Conv 3×3×128		
MP (2, 4) *(output: 48×341×128)*		
Conv 3×3×256		
MP (2, 4) *(output: 24×85×256)*		
Conv 3×3×512		
MP (2, 4) *(output: 12×21×512)*		
Conv 3×3×1024		
MP (3, 5) *(output: 4×4×1024)*		
Conv 3×3×2048		
MP (4, 4) *(output: 1×1×2048)*		
	Conv 1×1×1024	Conv 1×1×1024
	·	Conv 1×1×1024
Output 50×1 (sigmoid)		

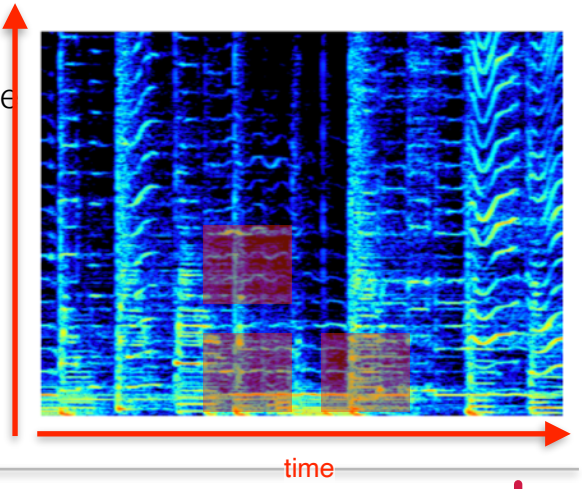
[Keunwoo Choi, Gyorgy Fazekas, and Mark Sandler. Automatic tagging using deep convolutional neural networks. In Proc. of ISMIR, 2016]

Input representation: 2D (time/frequency)

T/F images ≠ natural images ⇒ need to adapt the ConvNet architecture

– **Natural images**
  - (a) the two axis x and y
    – represent the same concept (spatial position)
  - (b) the elements of an image have
    – the same meaning independently of their positions over x and y.
  - (c) neighboring pixels:
    – usually highly correlated,
    – often belong to the same object

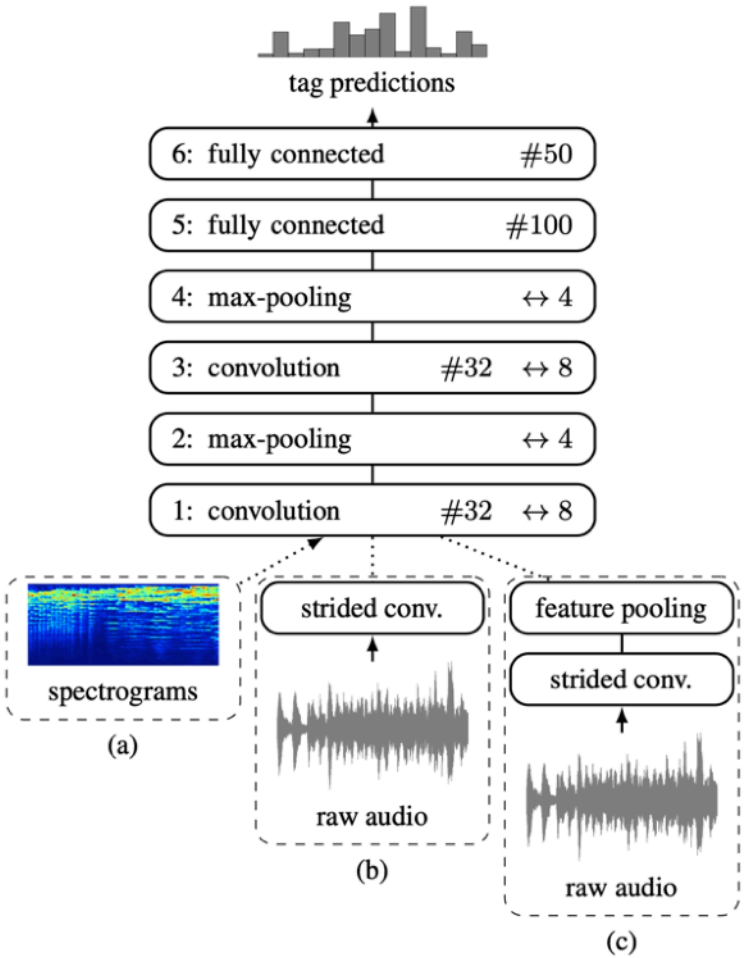
– **Time-frequency audio representations**
  - (a) the two axis x and y
    – represent profoundly different concepts (time and frequency).
  - (b) the elements of spectrogram (such as the T/F area of a source) have
    – the same meaning independently of its position over time
    – but not over frequency
      ⇒ no invariance over y, even in the case of log-frequencies
  - (c) neighboring pixels:
    – are not necessarily correlated
    – a given sound source (such has an harmonic sound) can be distributed over the whole frequency in a sparse way (the harmonics of a given sound can be spread over the whole



1D-Convolution on the audio waveform

– **2014 → Dieleman:**

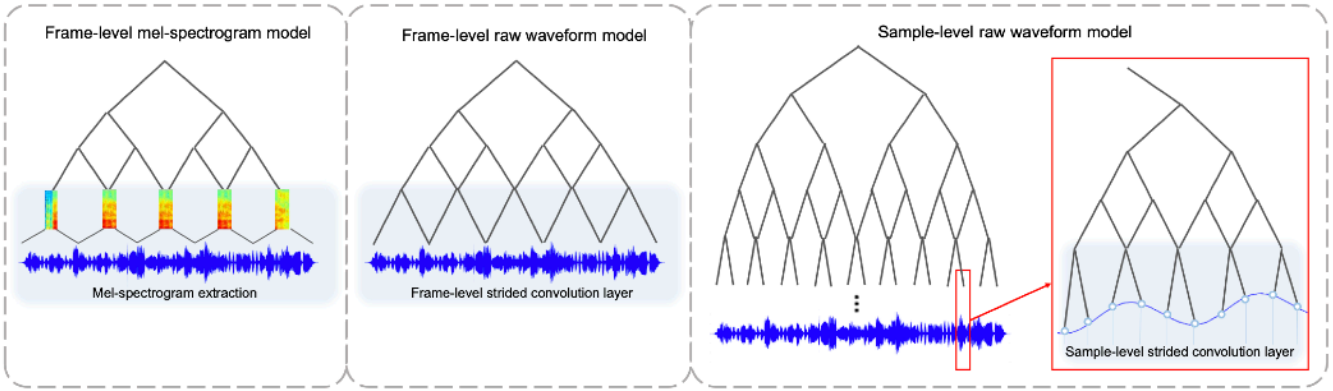
- **Task:** first end-to-end approaches for music auto-tagging,
- use 1D-convolution on the waveform to replace spectrogram input
- reproduce the spectrogram computation
  - spectrogram computed using a succession of DFTs computed with a frame length N and each separated by a hop size S
  - 1D-convolution with 1D-filters of length N and a stride of S
- **Results**: "end-to-end" approach under-performed the traditional spectrogram-based approach



[Sander Dieleman and Benjamin Schrauwen, "**End-to-end learning for music audio**", In IEEE ICASSP, 2014.]

2017 → Sample-CNN (1D-convolution on the audio waveform)

- **Idea**:
  - improve the Phase-invariance by reducing the size of the 1D-convolution filters (hence also the stride).
  - *If the filters are smaller, then the number of time translation to be learned is also reduced*
- **2017 → Sample-CNN**
  - a deep stack of 1D-convolution of small (3,1) filters applied to the waveform.
  - equivalent to the VGG-Net for 1D-convolution applied to waveforms.
  - Sample-CNN was shown to slightly outperforms the 2D-CNN on the spectrogram.



3^9 model, 19683 frames
59049 samples (2678 ms) as input

layer	stride	output	# of params
conv 3-128	3	19683 × 128	512
conv 3-128 maxpool 3	1 3	19683 × 128 6561 × 128	49280
conv 3-128 maxpool 3	1 3	6561 × 128 2187 × 128	49280
conv 3-256 maxpool 3	1 3	2187 × 256 729 × 256	98560
conv 3-256 maxpool 3	1 3	729 × 256 243 × 256	196864
conv 3-256 maxpool 3	1 3	243 × 256 81 × 256	196864
conv 3-256 maxpool 3	1 3	81 × 256 27 × 256	196864
conv 3-256 maxpool 3	1 3	27 × 256 9 × 256	196864
conv 3-256 maxpool 3	1 3	9 × 256 3 × 256	196864
conv 3-512 maxpool 3	1 3	3 × 512 1 × 512	393728
conv 1-512 dropout 0.5	1 –	1 × 512 1 × 512	262656
sigmoid	–	50	25650
Total params			1.9×10^6

[Jongpil Lee et al. "**Sample- level deep convolutional neural networks for music auto-tagging using raw waveforms**", 2017] LINK
[Taejun Kim et al. "**Sample-level CNN architectures for music auto-tagging using raw waveforms**". IEEE ICASSP 2018.] LINK

2018 → SincNet

- Problems of 1D-convolution
  - 1D-convolution filters are often difficult to interpret
  - Try to interpret the learned-filters un the frequency domain
- **SincNet:**
  - defines the 1D-filters as parametric functions $g(.)$ which theoretical frequency responses are parametrizable band pass filters
  - $g(.)$ is defined in the temporal domain as the difference between two $sinc(.)$ functions which learnable parameters define the low and high cutoff frequencies of the band-pass filters

$$y[n] = x[n] * g[n, \theta]$$

$$G[f, f_1, f_2] = rect\left(\frac{f}{2f_2}\right) - rect\left(\frac{f}{2f_1}\right),$$

$$g[n, f_1, f_2] = 2f_2 sinc(2\pi f_2 n) - 2f_1 sinc(2\pi f_1 n),$$

  - the filters obtained are much more interpretable
  - the performances for speaker recognition is much improved

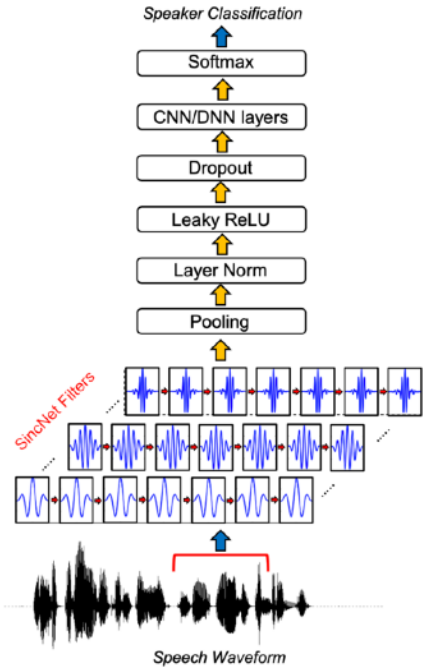


Fig. 1: Architecture of SincNet.

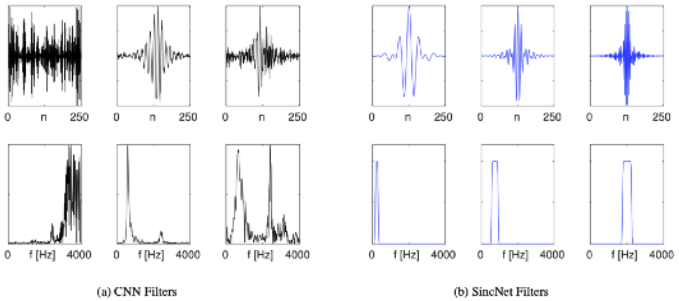


Fig. 2: Examples of filters learned by a standard CNN and by the proposed SincNet (using the Librispeech corpus). The first row reports the filters in the time domain, while the second one shows their magnitude frequency response.

[Mirco Ravanelli and Yoshua Bengio. "**Speaker recognition from raw waveform with SincNet**". In IEEE SLT, 2018] LINK

2021 → LEAF

- Learns all operations of audio features extraction
  - from filtering to pooling, compression and normalization
- **Filtering**:
  - Gabor filters frequencies η_n, bandwidth σ_n $\varphi_n(t) = e^{i2\pi\eta_n t}\frac{1}{\sqrt{2\pi}\sigma_n}e^{-\frac{t^2}{2\sigma_n^2}}, \quad n = 1,\dots,N, \quad t = -W/2,\dots,W/2.$
- **Pooling**:
  - Learnable low-pass pooling $\phi_n(t) = \frac{1}{\sqrt{2\pi}\sigma_n}e^{-\frac{t^2}{2\sigma_n^2}}, \quad t = -W/2,\dots,W/2.$
- **Compression and normalization** for each channel n independently
  - PCEN (Per-Channel Energy Normalization): $\mathrm{PCEN}(\mathcal{F}(t,n)) = \left(\frac{\mathcal{F}(t,n)}{(\varepsilon + \mathcal{M}(t,n))^{\alpha_n}} + \delta_n\right)^{r_n} - \delta_n^{r_n},$
  - $\mathcal{M}$ exponential moving average over time
  - Compression r_n

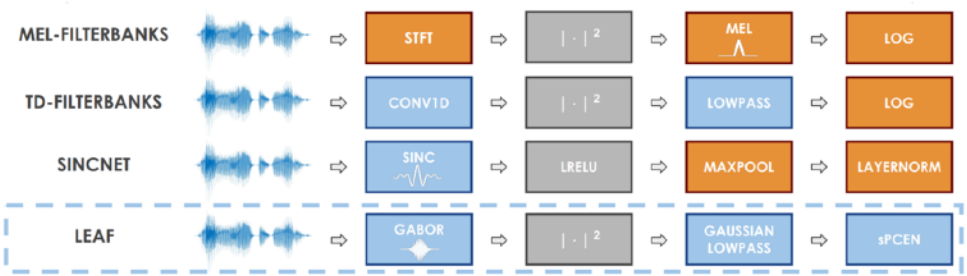


Table 1: Test accuracy (%) for single-task classification.

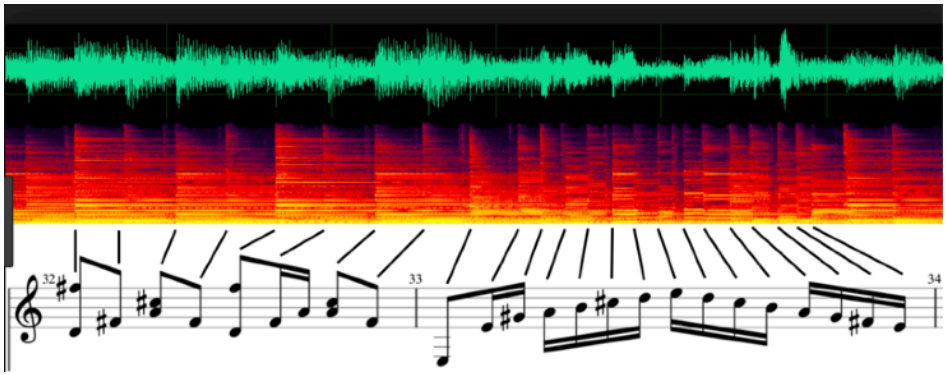
Task	Mel	TD-fbanks	SincNet	LEAF
Acoustic scenes	99.2 ± 0.4	$\mathbf{99.5} \pm 0.3$	96.7 ± 0.9	99.1 ± 0.5
Birdsong detection	78.6 ± 1.0	80.9 ± 0.9	78.0 ± 1.0	$\mathbf{81.4} \pm 0.9$
Emotion recognition	49.1 ± 2.4	57.1 ± 2.4	44.2 ± 2.4	$\mathbf{57.8} \pm 2.4$
Speaker Id. (VC)	31.9 ± 0.7	25.3 ± 0.7	$\mathbf{43.5} \pm 0.8$	33.1 ± 0.7
Music (instrument)	$\mathbf{72.0} \pm 0.6$	70.0 ± 0.6	70.3 ± 0.6	$\mathbf{72.0} \pm 0.6$
Music (pitch)	91.5 ± 0.3	91.3 ± 0.3	83.8 ± 0.5	$\mathbf{92.0} \pm 0.3$
Speech commands	92.4 ± 0.4	87.3 ± 0.4	89.2 ± 0.4	$\mathbf{93.4} \pm 0.3$
Language Id.	76.5 ± 0.4	71.6 ± 0.5	78.9 ± 0.4	$\mathbf{86.0} \pm 0.4$
Average	73.9 ± 0.8	72.9 ± 0.8	73.1 ± 0.9	$\mathbf{76.9} \pm 0.8$

[Neil Zeghidour et al., "**LEAF: A Learnable Frontend for Audio Classification**", in ICLR, 2021] LINK

2017 → Music Multiple-f_0 Estimation using Harmonic-CQT

Introduction

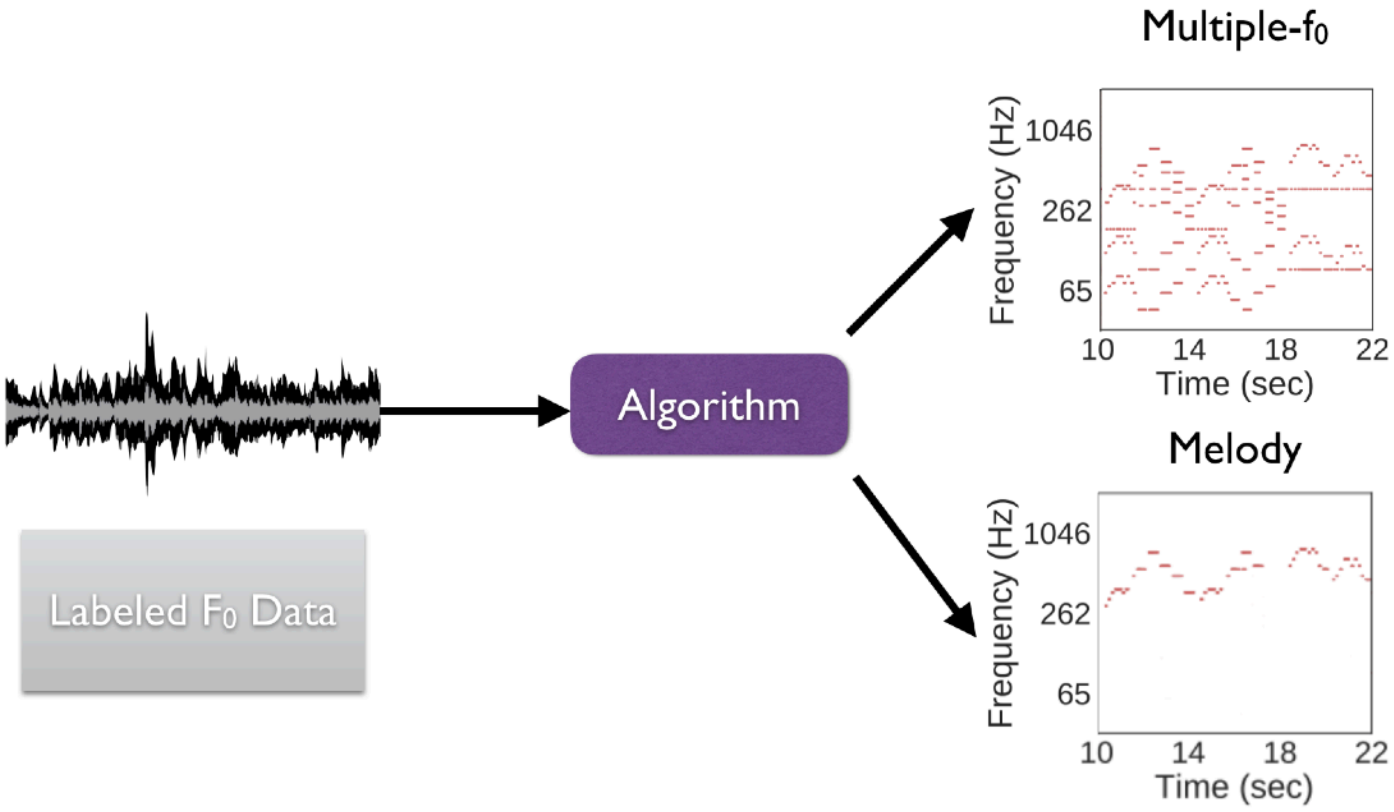
- **Goal** :
  - obtain a transcription of the content of an audio signal
    - which notes ? which chords ? which instruments ?
    - equivalent to Automatic Speech Recognition
  - difficulty :
    - all instruments are super-imposed in the audio signal
  - sub-tasks :
    - only transcribe the notes / f0 (in Hz)
    - only transcribe the dominant melody
- **How can we do that ?**
  - many methods proposed in the past : sinusoidal model, NMF, PLCA, ... ,
  - large improvement since 2017 using ... deep learning



[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

2017 → Music Multiple-f_0 Estimation using Harmonic-CQT

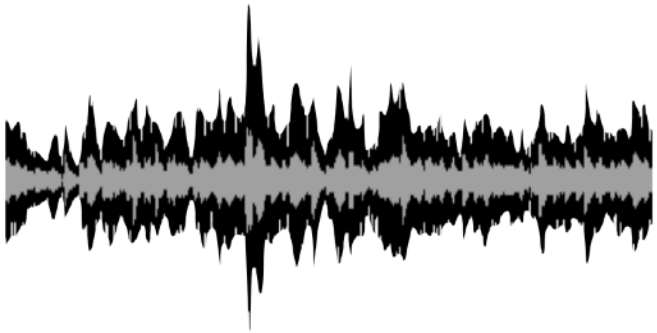
Introduction

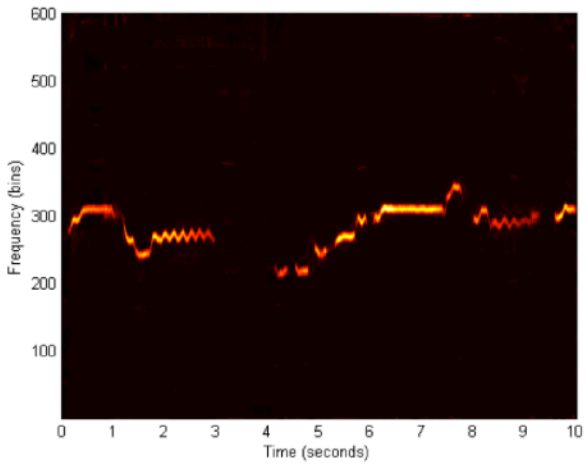


[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

2017 → Music Multiple-f_0 Estimation using Harmonic-CQT

Learning a "Saliency" Representation



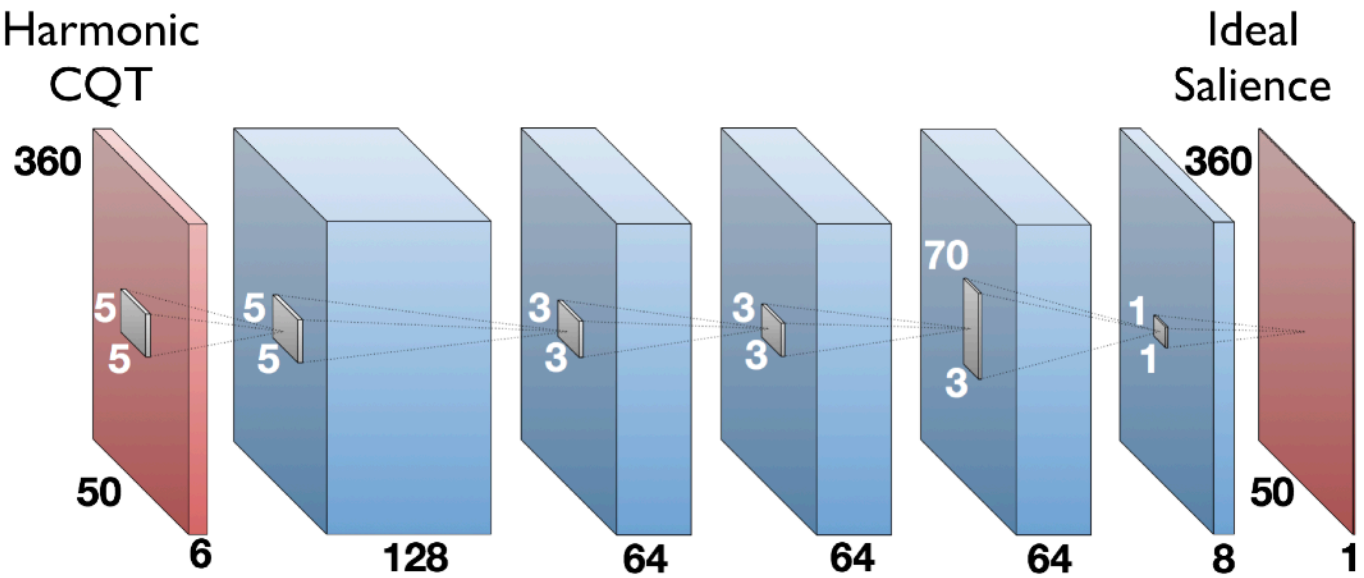


[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

2017 → Music Multiple-f_0 Estimation using Harmonic-CQT

Using Convolutional Neural Network

- **Inputs** :
  - magnitude spectrogram **with depth** (Harmonic CQT)
- **Outputs** :
  - image of the same size of input but with only saliency information

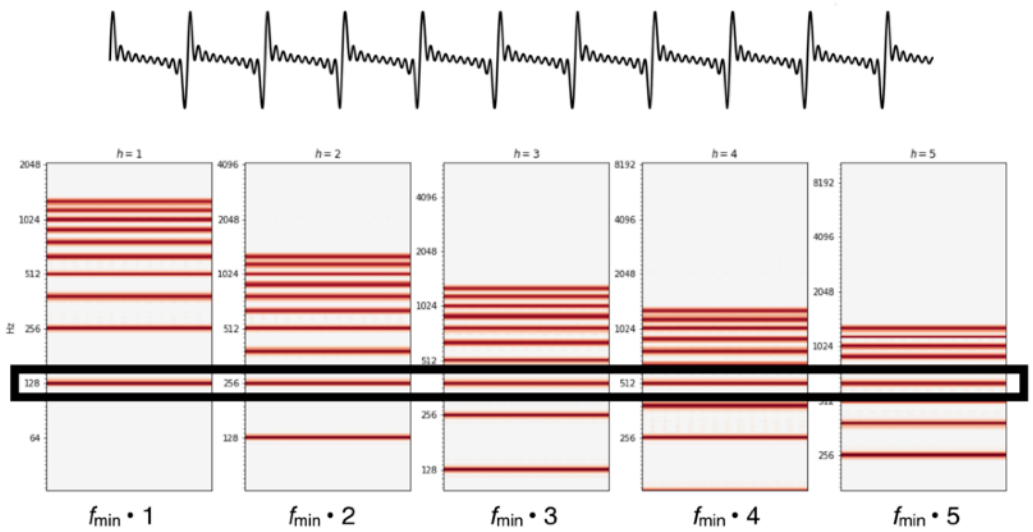


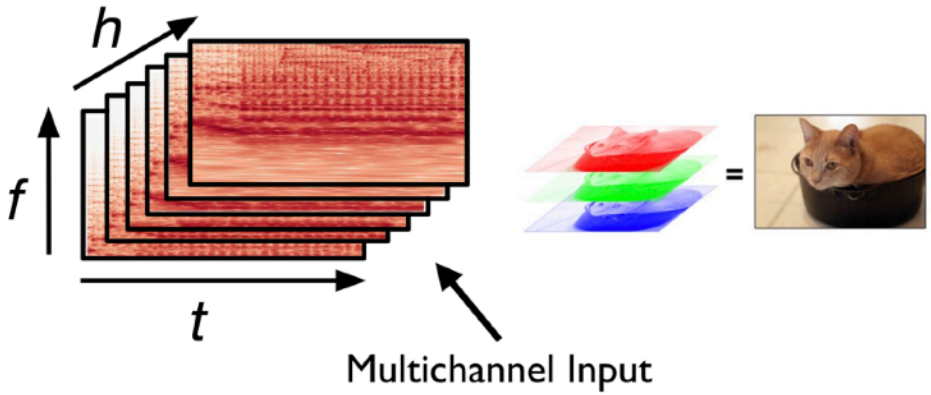
$$L(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

2017 → Music Multiple-f_0 Estimation using Harmonic-CQT

Input Representation: a Harmonic CQT

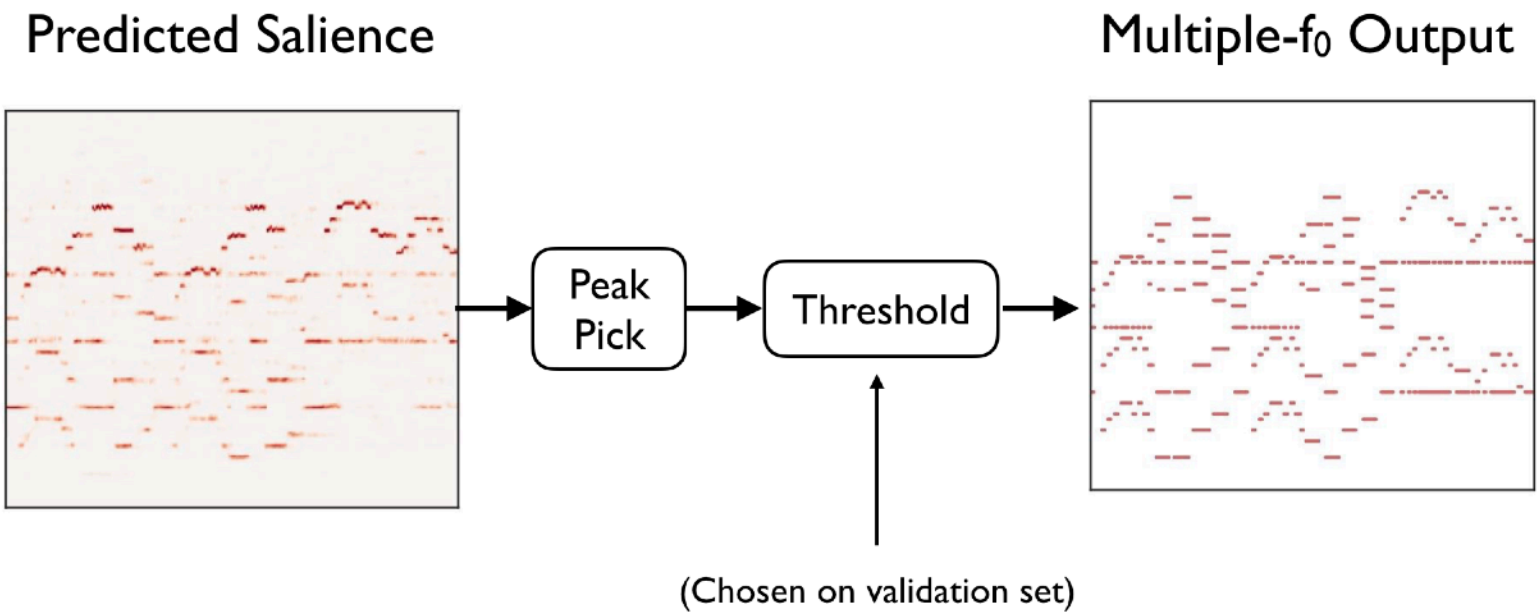




[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

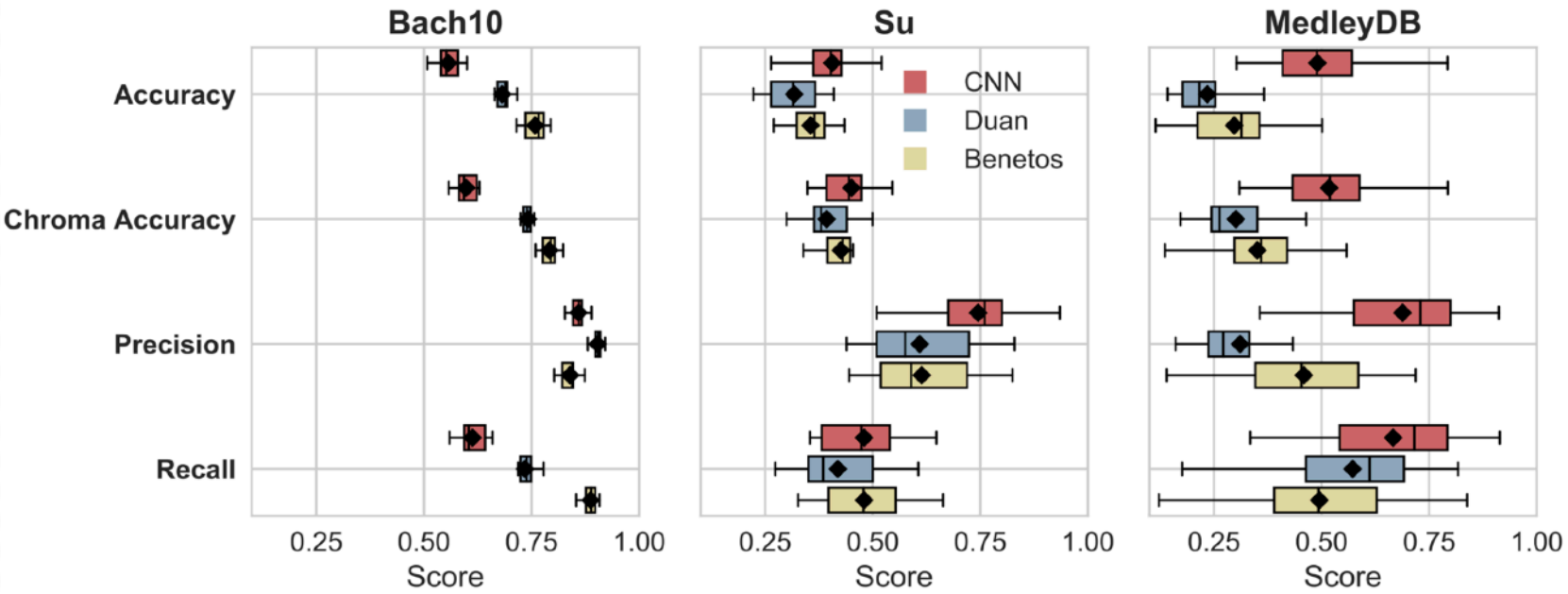
2017 → Music Multiple-f_0 Estimation using Harmonic-CQT

Multiple-f_0 Estimation



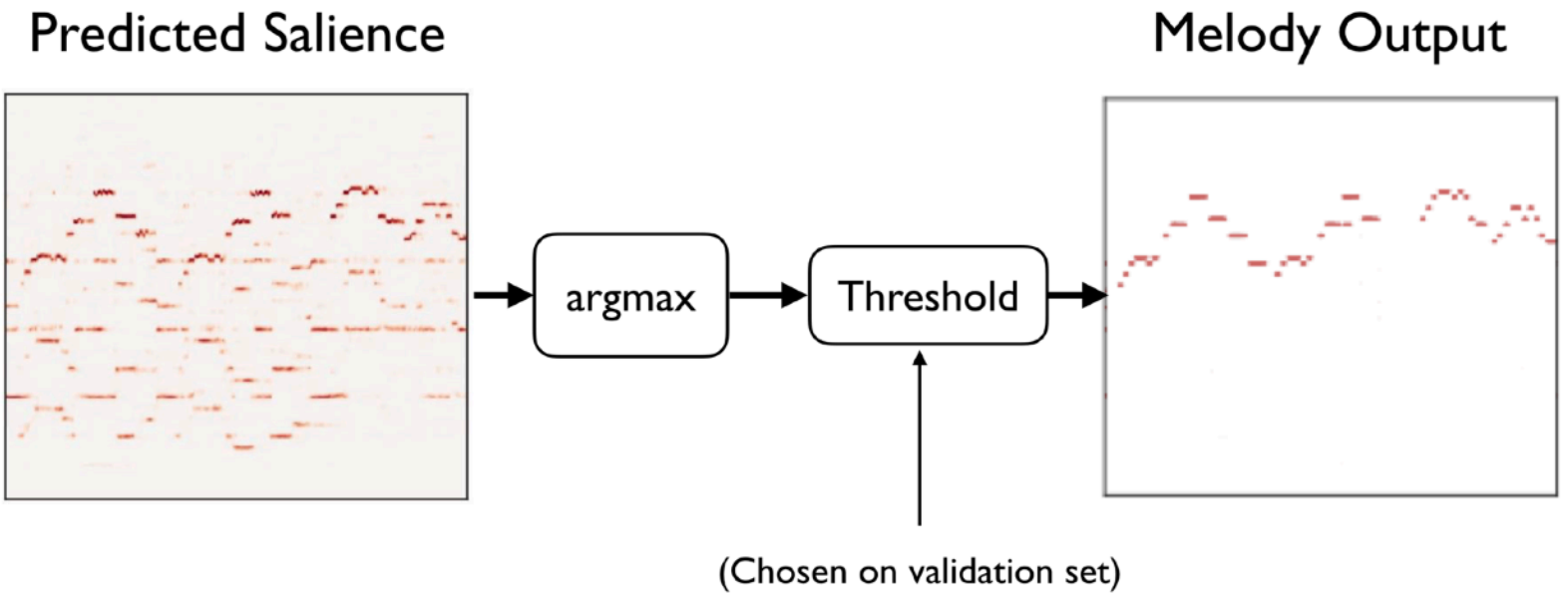
[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

Multiple-f_0 Estimation: Results



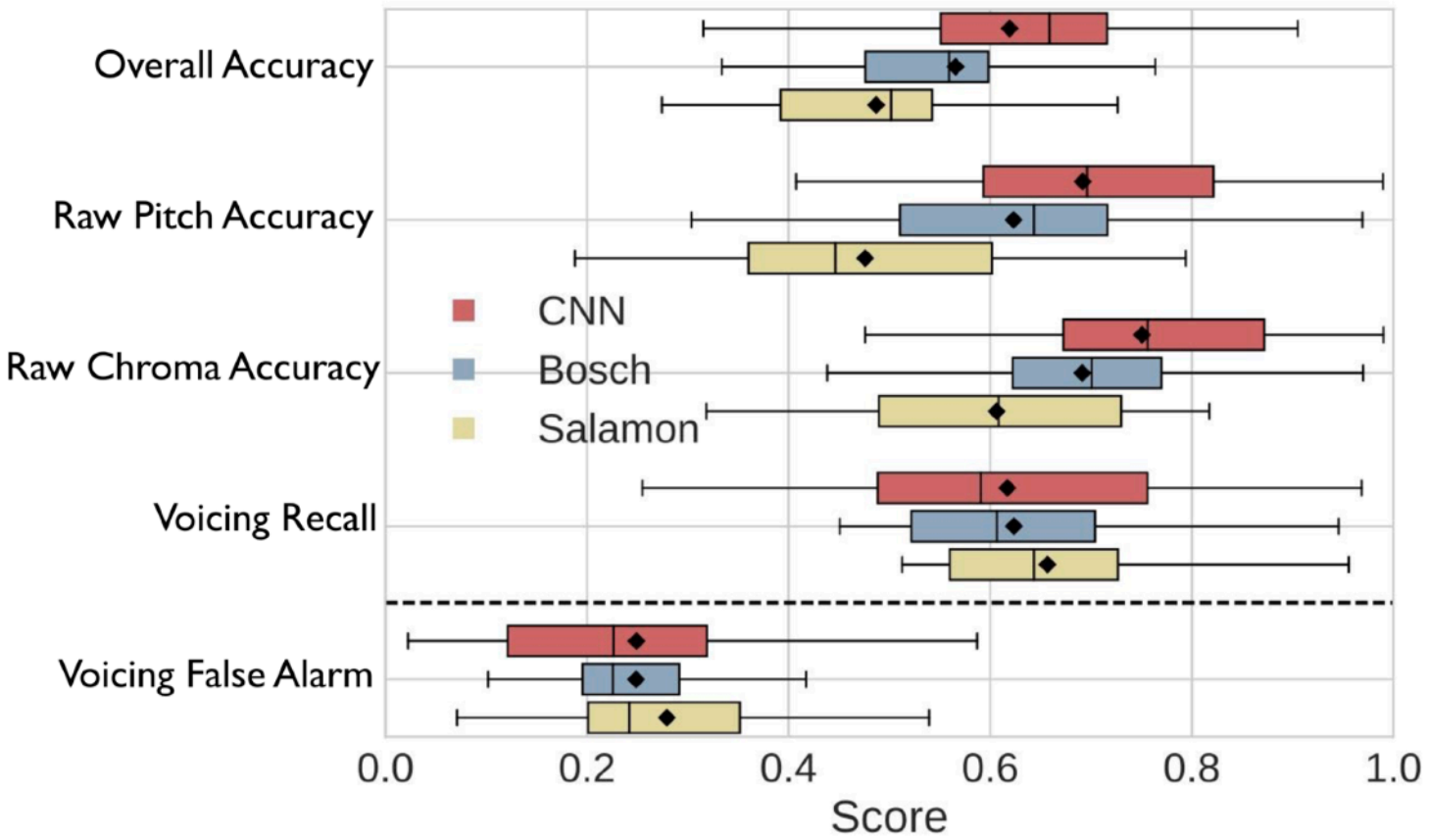
[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

Dominant Melody Estimation



[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

Dominant Melody Estimation: Results



[R. Bittner et al. "**Deep salience representations for f0 estimation in polyphonic music**", in ISMIR, 2017] LINK

Example code
H-CQT

- https://colab.research.google.com/drive/1TVHnIJ5kv1pyua-KNFjR0zqiqdsiKiJ_

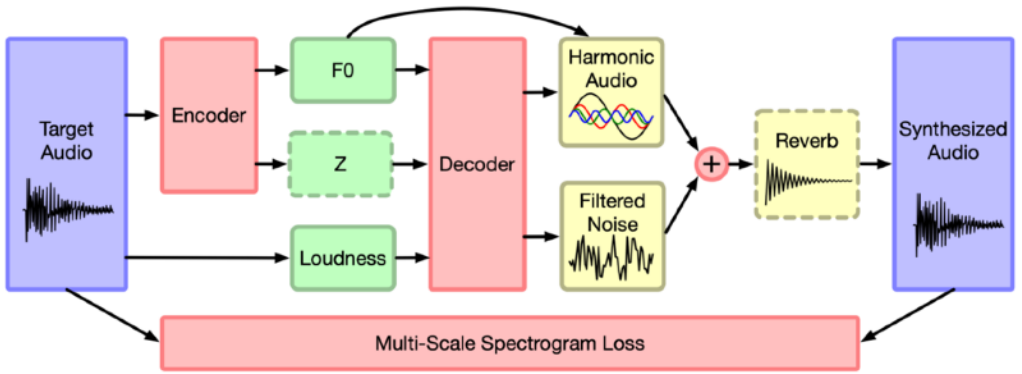
2020 → Differentiable Digital Signal Processing (DDSP)

- DDSP defines the sound production model:
  - the **Spectral Modelling Synthesis (SMS) model [Serra,1990]**
    - combines harmonic additive synthesis (adding together many harmonic sinusoidal components) with subtractive synthesis (filtering white noise);
    - also adds room acoustics to the produced sound through reverberation

$$x(n) = \sum_{k=1}^{K} A_k(n) \cdot \sin(\phi_k(n)) \qquad \phi_k(n) = 2\pi \sum_{m=0}^{n} f_k(m) + \phi_{0,k} \qquad f_k(n) = k f_0(n)$$

- The training consists in finding its parameters !

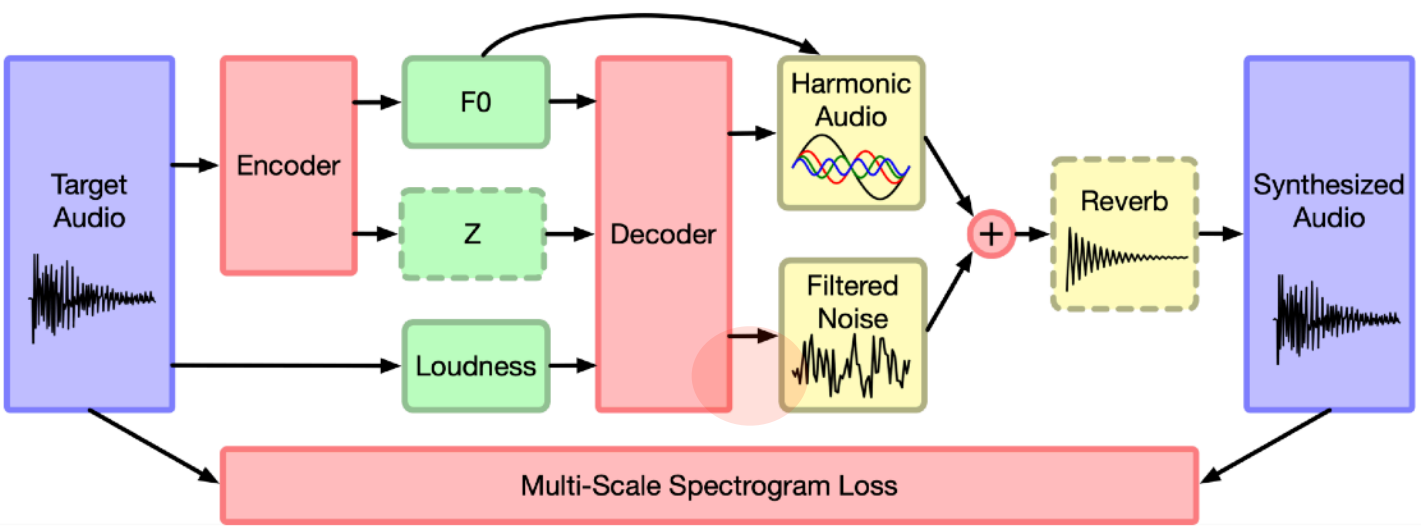
- Input audio signal $\mathbf{x}(t)$
  - first encoded into time-varying
    - loudness $l(t)$,
    - pitch $f_0(t)$
    - a latent representation $\mathbf{z}(t)$
- Those are fed to a decoder
  - which estimates the control parameters of the additive and filtered noise synthesizers



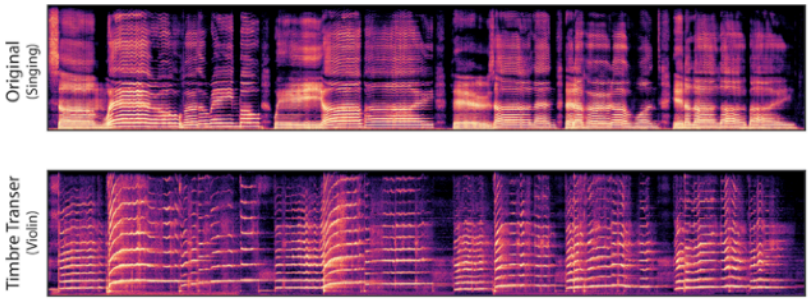
[J. Engel, L. Hantrakul, C. Gu, and A. Roberts. "**DDSP: Differentiable digital signal processing**". In ICLR, 2020] LINK

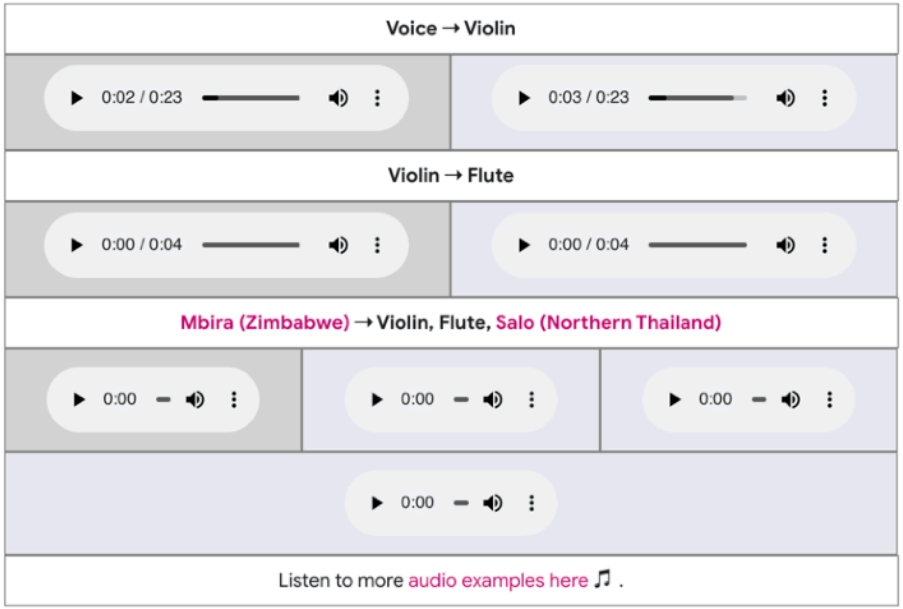
2020 → Differentiable Digital Signal Processing (DDSP)

- **Encoder:**
  - Loudness: extracted directly from the audio
  - f_0 pitch: Crepe or ResNet
  - $z(t)$: time-varying latent encoding
- **Decoder:**
  - map $(f(t), l(t), z(t))$to control parameters additive and filtered noise synthesizers



2020 → Differentiable Digital Signal Processing (DDSP)





https://magenta.tensorflow.org/ddsp

Links

- **Signal processing Tools**:
  - https://librosa.org/doc/latest/index.html
- **Speech tools**
  - https://kaldi-asr.org/
  - https://htk.eng.cam.ac.uk/
  - https://speechbrain.github.io/
- **MIR tools:**
  - http://ismir.net
- **DCASE tools:**
  - http://dcase.community
- **Conferences**:
  - icassp, interspeech, eusipco, iclr, neurips
- **Books**
  - G. Peeters and G. Richard. "Deep learning for audio and music" In "Multi-faceted Deep Learning: Models and Data, chapter 11", 2021.
  - M. Müller, "Fundamentals of Music Processing: Using Python and Jupyter Notebooks 2nd ed", 2021