Randomly weighted CNNs for audio classification: a personal (re)view

Jordi Pons

jordipons.me – @jordiponsdotme

Music Technology Group
Universitat Pompeu Fabra, Barcelona

Outline

CNN architectures for audio classification: a review

Randomly weighted CNNs: how these work in practice?

Acronyms

MLP: multi layer perceptron ≡ feed-forward neural network
RNN: recurrent neural network
LSTM: long-short term memory
CNN: convolutional neural network
BN: batch normalization

..the following slides assume you know these concepts!

Outline

CNN architectures for audio classification: a review

Randomly weighted CNNs: how these work in practice?

Which is our goal / task?

input → machine learning → output

waveform

or any audio representation!

deep learning model

phonetic transcription

(music) audio tagging

event detection

The deep learning pipeline

input → front-end → back-end → output

waveform

or any audio representation!

phonetic transcription

(music) audio tagging

event detection

The deep learning pipeline: input?

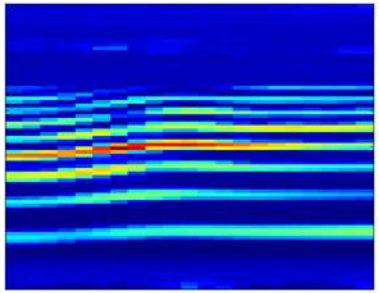
input →

?

How to format the input (audio) data?

Waveform
end-to-end learning

Time-frequency representation
e.g.: log-mel spectrogram



The deep learning pipeline: front-end?

input → front-end

waveform

spectrogram

?

based on domain knowledge?	**filters config?**	**input signal?**	
		waveform	*spectrogram*

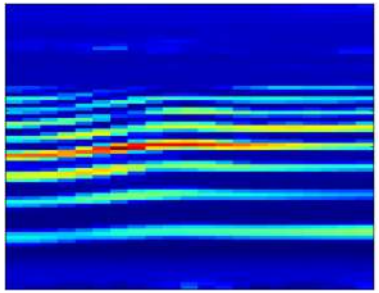
CNN front-ends for audio classification

Waveform
end-to-end learning

3x1 | 3x1 | ... | 3x1 | 3x1

Sample-level

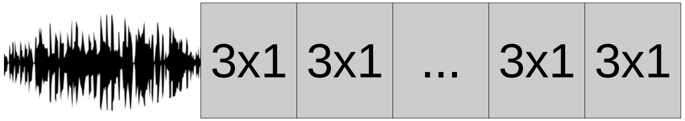
Time-frequency representation
e.g.: log-mel spectrogram

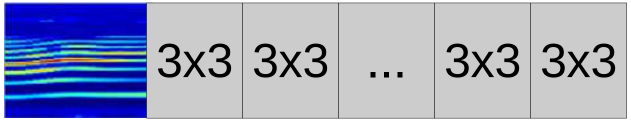


3x3 | 3x3 | ... | 3x3 | 3x3

Small-rectangular filters

based on domain knowledge?	filters config?	input signal?	
		waveform	*spectrogram*
no	*minimal* *filter expression*	sample-level	small-rectangular filters

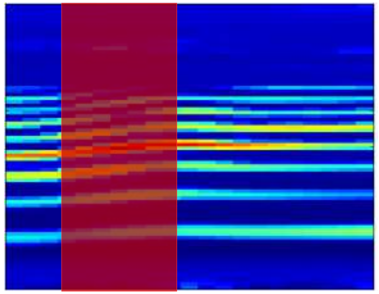




Domain knowledge to design CNN front-ends

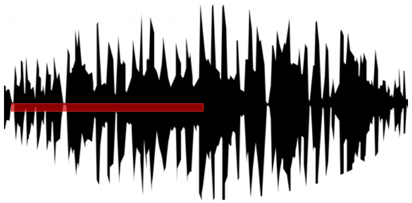
Waveform
end-to-end learning

Time-frequency representation
e.g.: log-mel spectrogram



Domain knowledge to design CNN front-ends

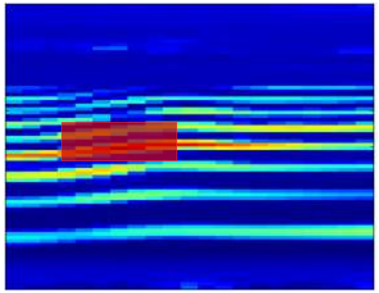
Waveform
end-to-end learning



filter length: 512 ***window length?***
stride: 256 ***hop size?***

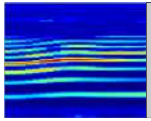
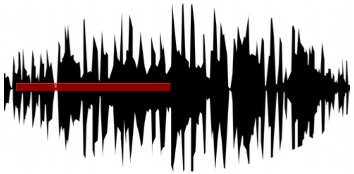
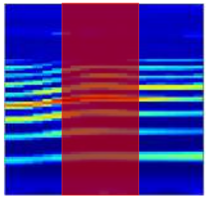
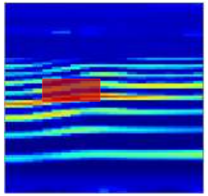
frame-level

Time-frequency representation
e.g.: log-mel spectrogram



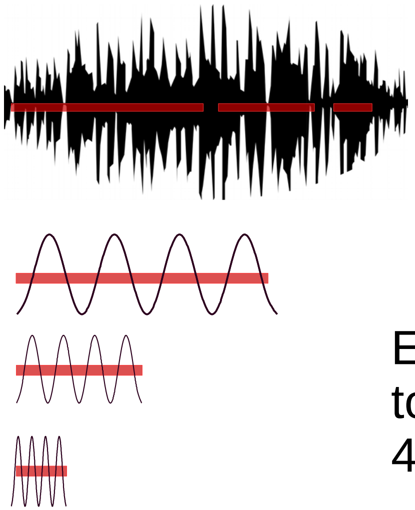
Explicitly tailoring the CNN towards learning temporal ***or*** timbral cues

vertical or horizontal filters

based on domain knowledge?	filters config?	input signal? *waveform*	input signal? *spectrogram*
no	*minimal filter expression*	sample-level  3x1 | 3x1 | ... | 3x1 | 3x1	small-rectangular filters  3x3 | 3x3 | ... | 3x3 | 3x3
yes	*single filter shape in 1st CNN layer*	frame-level 	vertical *OR* horizontal  *or* 

DSP wisdom to design CNN front ends

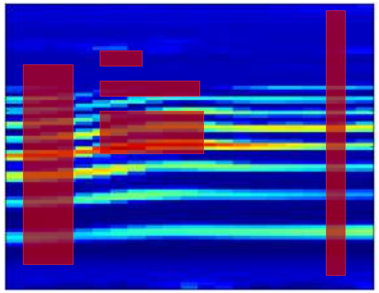
Waveform
end-to-end learning



Efficient way
to represent
4 periods!

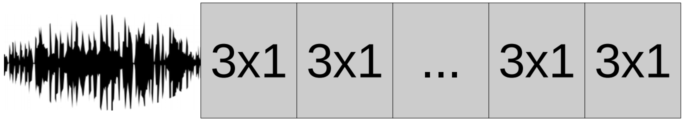
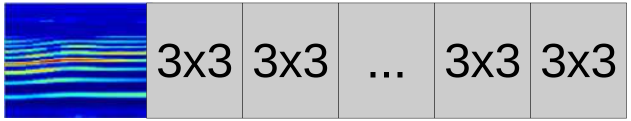
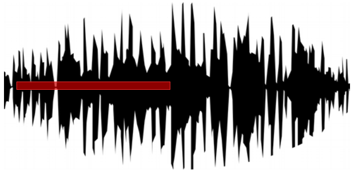
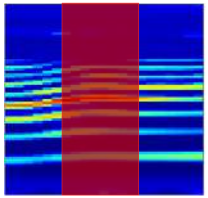
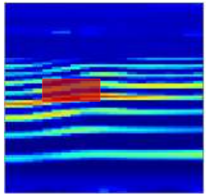
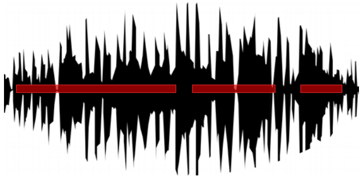
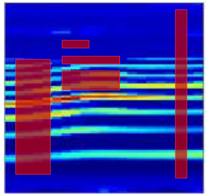
Frame-level (many shapes!)

Time-frequency representation
e.g.: log-mel spectrogram



Explicitly tailoring the CNN towards
learning temporal ***and*** timbral cues

Vertical and/or horizontal

based on domain knowledge?	**filters config?**	**input signal?** *waveform*	**input signal?** *spectrogram*
no	*minimal* filter expression	sample-level 	small-rectangular filters 
yes	*single filter shape in 1st CNN layer*	frame-level 	vertical *OR* horizontal  *or* 
yes	*many filter shapes in 1st CNN layer*	frame-level 	vertical *AND/OR* horizontal 

CNN front-ends for audio classification

Sample-level: Lee et al., 2017 – **Sample-level Deep Convolutional Neural Networks for Music Auto-tagging Using Raw Waveforms** *in Sound and Music Computing Conference (SMC)*

Small-rectangular filters: Choi et al., 2016 – **Automatic tagging using deep convolutional neural networks** *in Proceedings of the ISMIR (International Society of Music Information Retrieval) Conference*

Frame-level (single shape): Dieleman et al., 2014 – **End-to-end learning for music audio** *in International Conference on Acoustics, Speech and Signal Processing (ICASSP)*

Vertical: Lee et al., 2009 – **Unsupervised feature learning for audio classification using convolutional deep belief networks** *in Advances in Neural Information Processing Systems (NIPS)*

Horizontal: Schluter & Bock, 2014 – **Improved musical onset detection with convolutional neural networks** *in International Conference on Acoustics, Speech and Signal Processing (ICASSP)*

Frame-level (many shapes): Zhu et al., 2016 – **Learning multiscale features directly from waveforms** *in arXiv:1603.09509*

Vertical and horizontal (many shapes): Pons, et al., 2016 – **Experimenting with musically motivated convolutional neural networks** *in 14th International Workshop on Content-Based Multimedia Indexing*

The deep learning pipeline: output

input → front-end → back-end → output

waveform

spectrogram

several CNN architectures

DNN?
RNN?
Attention?

phonetic transcription

(music) audio tagging

event detection

The deep learning pipeline: output

input	front-end	back-end	output
waveform spectrogram	*several CNN architectures*	*DNN?* *RNN?* *Attention?*	phonetic transcription **(music) audio tagging** event detection

Outline

CNN architectures for audio classification: a review

Randomly weighted CNNs: how these work in practice?

ArXiv: *https://arxiv.org/abs/1805.00237*
Code: *https://github.com/jordipons/elmarc*

Methodology

On Random Weights and Unsupervised Feature Learning

Andrew M. Saxe, Pang Wei Koh, Zhenghao Chen,
Maneesh Bhand, Bipin Suresh, and Andrew Y. Ng
Stanford University
Stanford, CA 94305
{asaxe,pangwei,zhenghao,mbhand,bipins,ang}@cs.stanford.edu

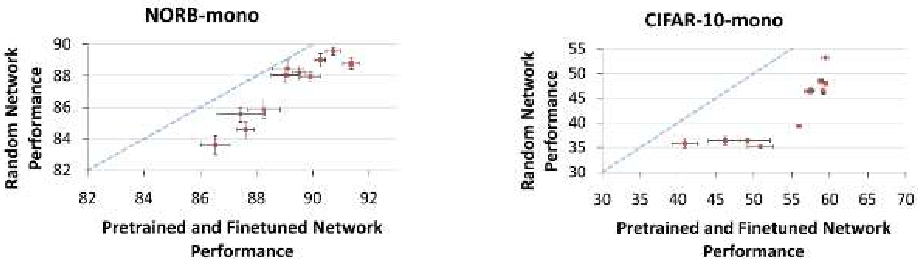


Figure 5: Classification performance of random-weight networks vs pretrained and finetuned networks. Left: NORB-mono. Right: CIFAR-10-mono (Error bars represent a 95% confidence interval about the mean)

4 Fast architecture selection

When we plot the classification performance of random-weight architectures against trained-weight architectures, a distinctive trend emerges: we see that architectures which perform well with random weights also tend to perform well with pretrained and finetuned weights, and vice versa (Fig. 5). Intuitively, our analysis in Section 2 suggests that random-weight performance is not truly random but should correlate with the corresponding trained-weight performance, as both are linked to intrinsic properties of the architecture. Indeed, this happens in practice.

Methodology

Goal? Compare different (randomly weighted) architectures

Method? Features (embeddings of random CNN)
+ classifier

Compare classification accuracies when
using different (randomly weighted) architectures

Data? Fault-filtered GTZAN,
Extended Ballroom, UrbanSounds8k

Pipeline of our study: input?

input → front-end → classifier → output

waveform

log-mel
spectrogram

(music) audio
tagging

Pipeline of our study: front-end?

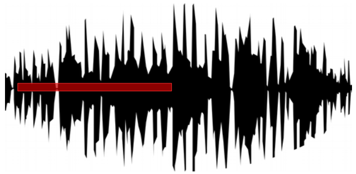
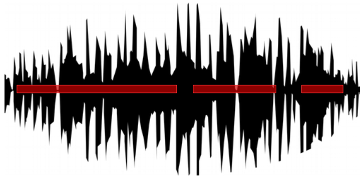
input → front-end → classifier → output

waveform

log-mel
spectrogram

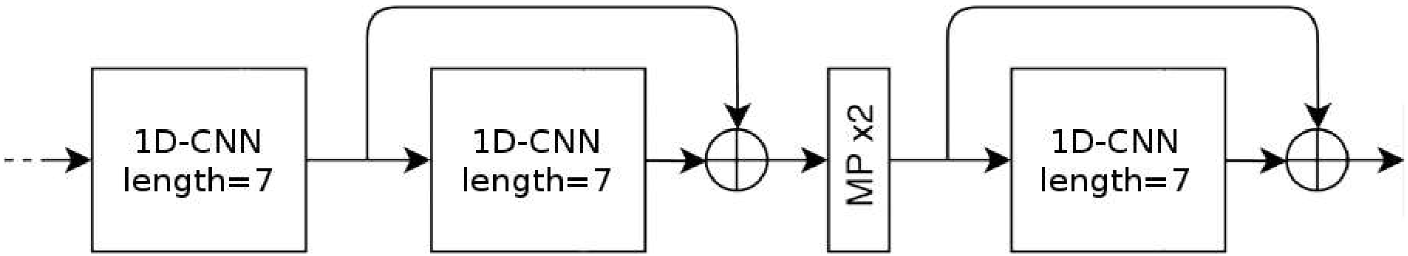
?

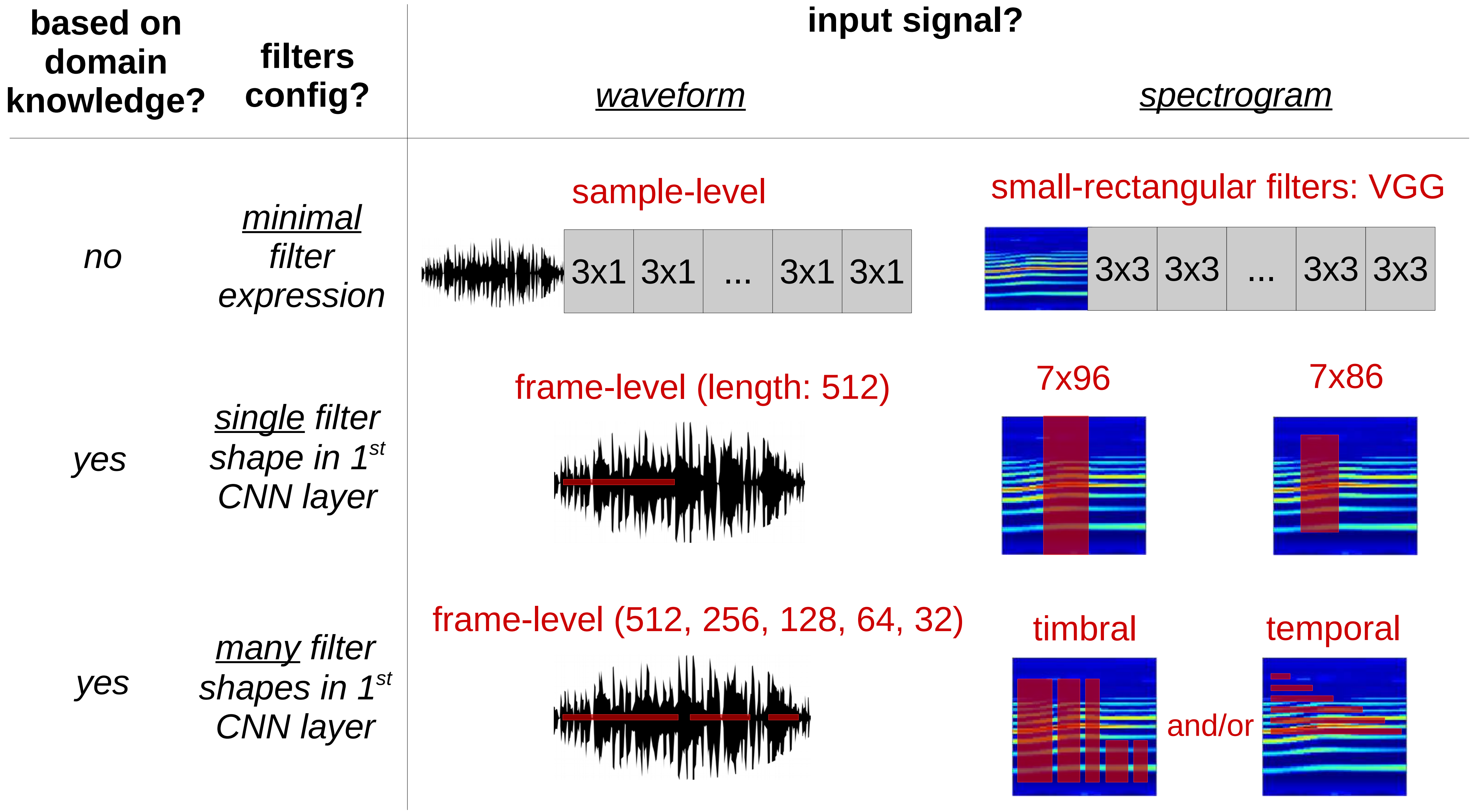
(music) audio
tagging

based on domain knowledge?	**filters config?**	**input signal?** *waveform*	*spectrogram*
no	*minimal* filter expression	sample-level  3x1 | 3x1 | ... | 3x1 | 3x1	
yes	*single* filter shape in 1st CNN layer	frame-level (length: 512) 	
yes	*many* filter shapes in 1st CNN layer	frame-level (512, 256, 128, 64, 32) 	

Architectural details: waveform models

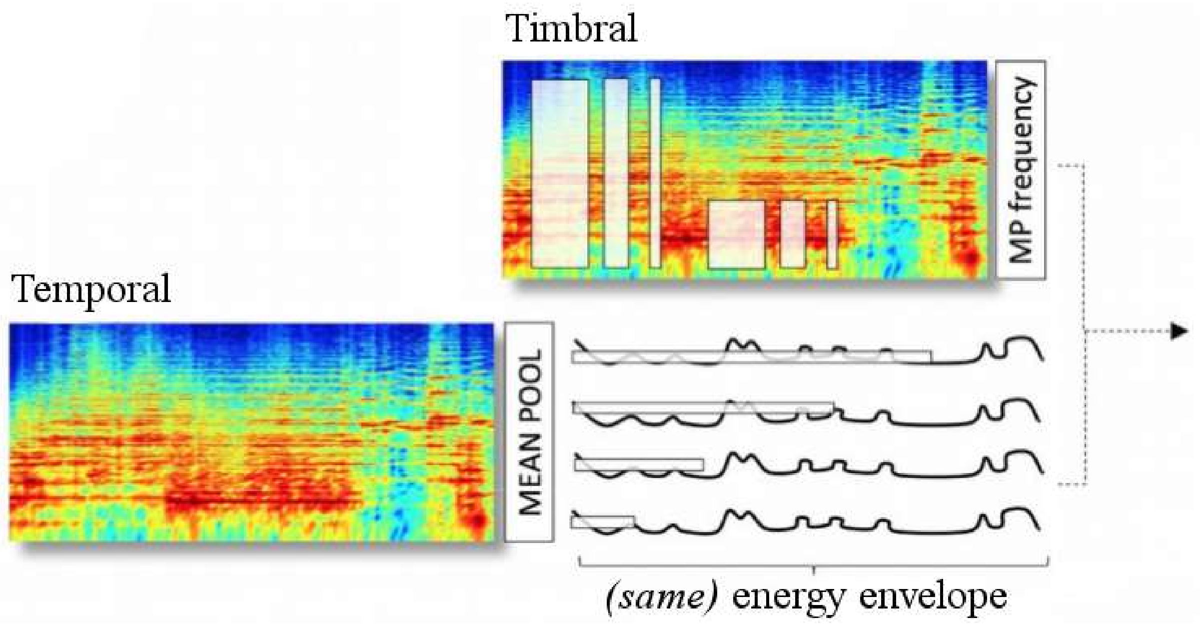
Additional layer for the fame-level architectures to
allow a fair comparison with the (deep) sample-level



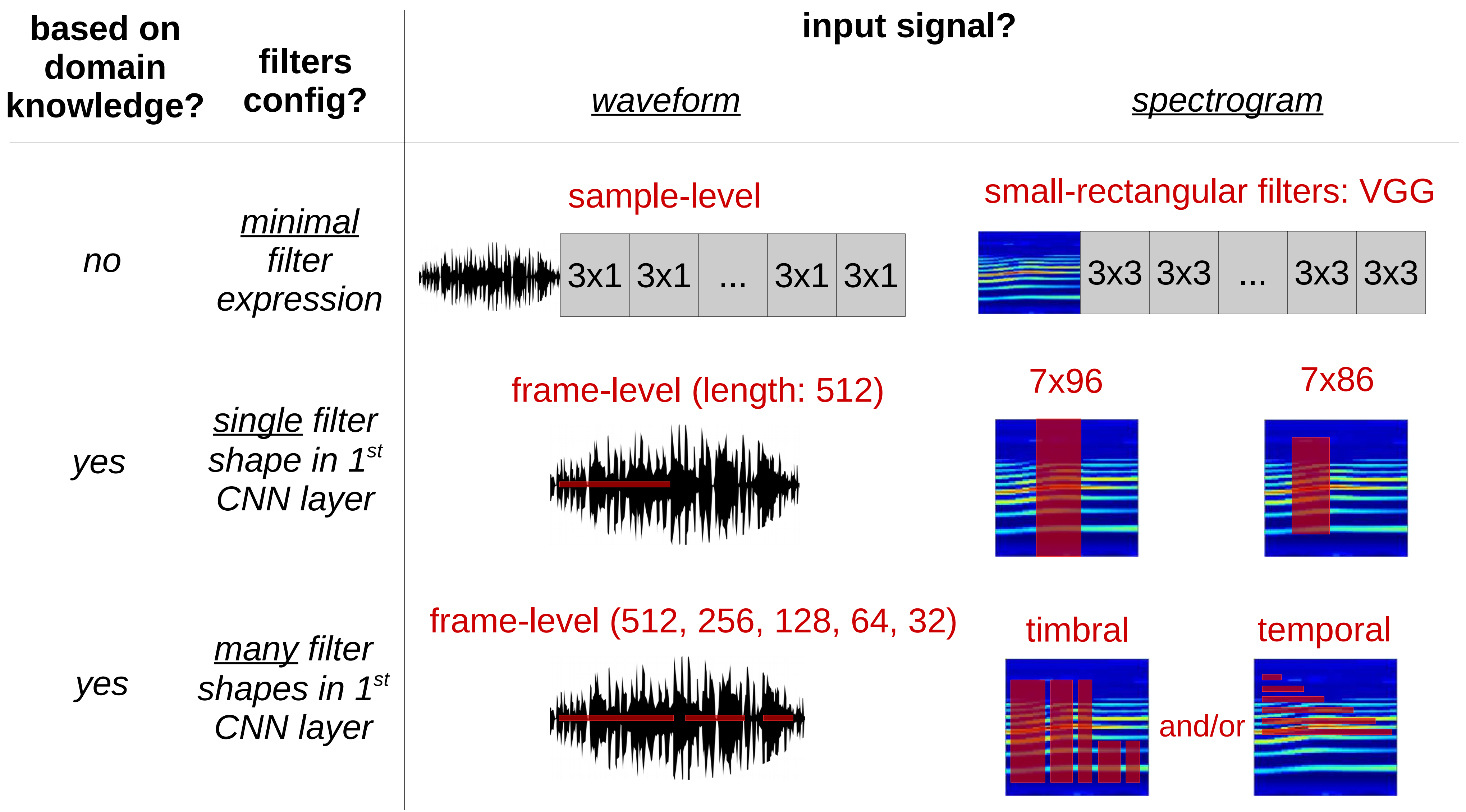


based on domain knowledge?
filters config?
input signal?
waveform
spectrogram
no
minimal filter expression
sample-level
3x1 3x1 ... 3x1 3x1
small-rectangular filters: VGG
3x3 3x3 ... 3x3 3x3
yes
single filter shape in 1st CNN layer
frame-level (length: 512)
7x96
7x86
yes
many filter shapes in 1st CNN layer
frame-level (512, 256, 128, 64, 32)
timbral
and/or
temporal

Further details about the timbral + temporal



Musically motivated CNNs
(Pons et al., 2016 – 2017)


based on domain knowledge?
filters config?
input signal?
waveform
spectrogram
no
minimal filter expression
sample-level
3x1 3x1 ... 3x1 3x1
small-rectangular filters: VGG
3x3 3x3 ... 3x3 3x3
yes
single filter shape in 1st CNN layer
frame-level (length: 512)
7x96
7x86
yes
many filter shapes in 1st CNN layer
frame-level (512, 256, 128, 64, 32)
timbral
and/or
temporal

The deep learning pipeline: back-end?

input → front-end → classifier → output

waveform

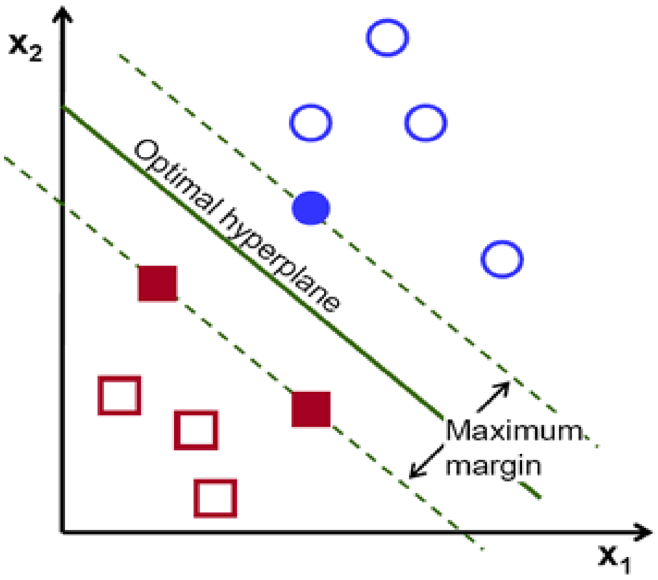
log-mel spectrogram

nine randomly weighted CNN architectures

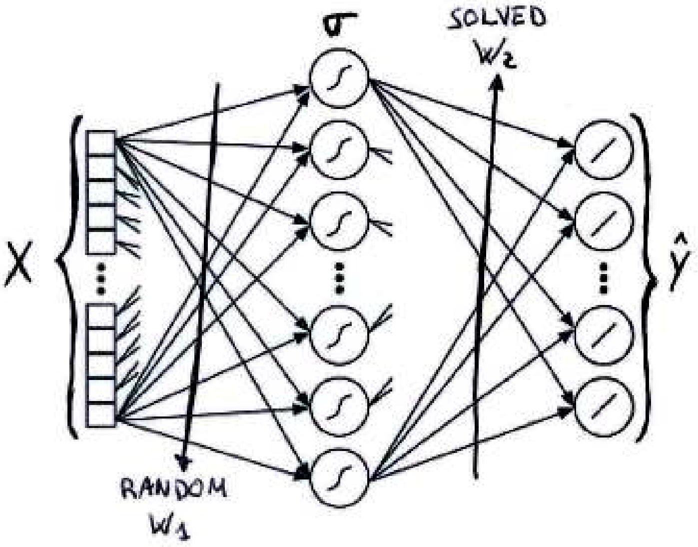
?

(music) audio tagging

Studied back-ends: SVM and ELM classifiers



SVM: support vector machine

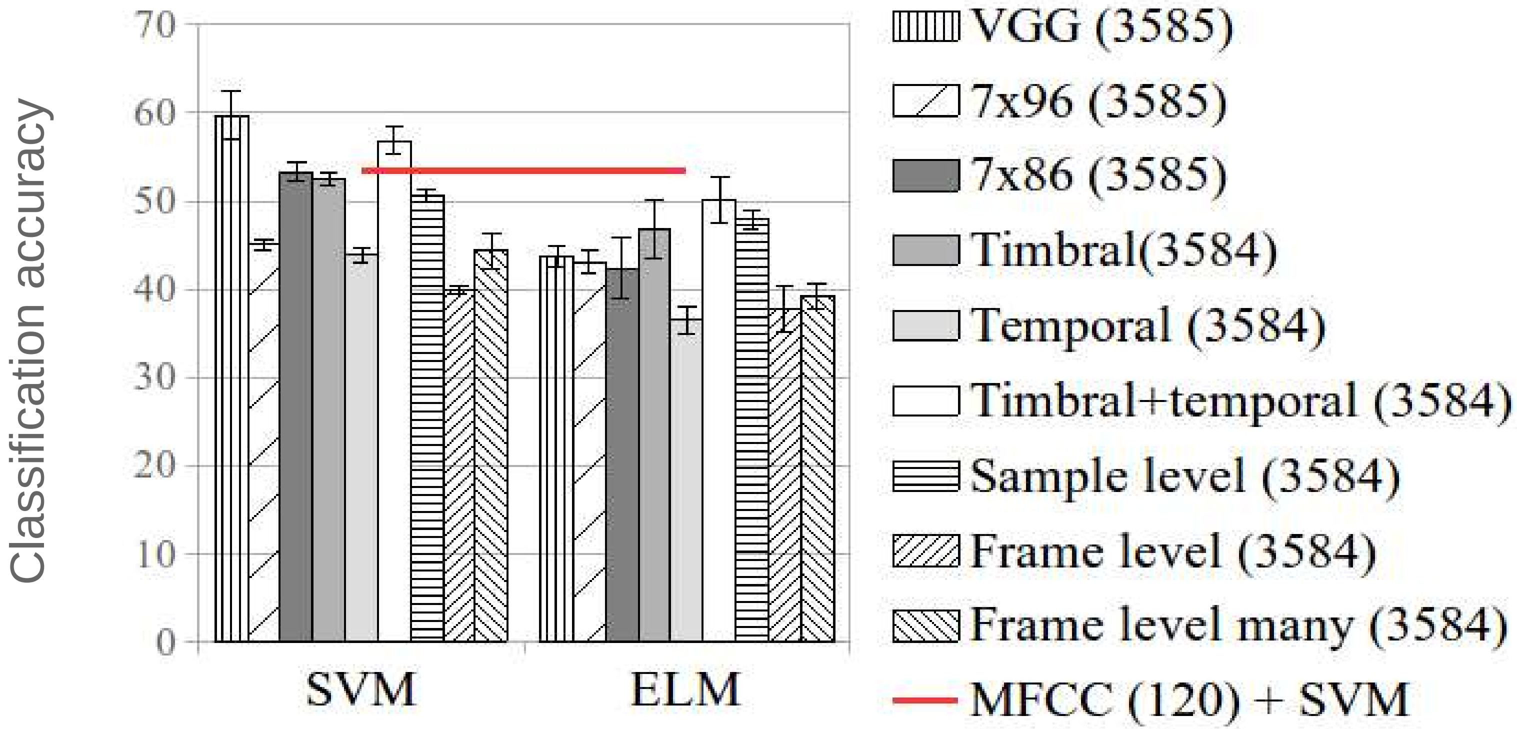


ELM: extreme learning machine

The deep learning pipeline: output

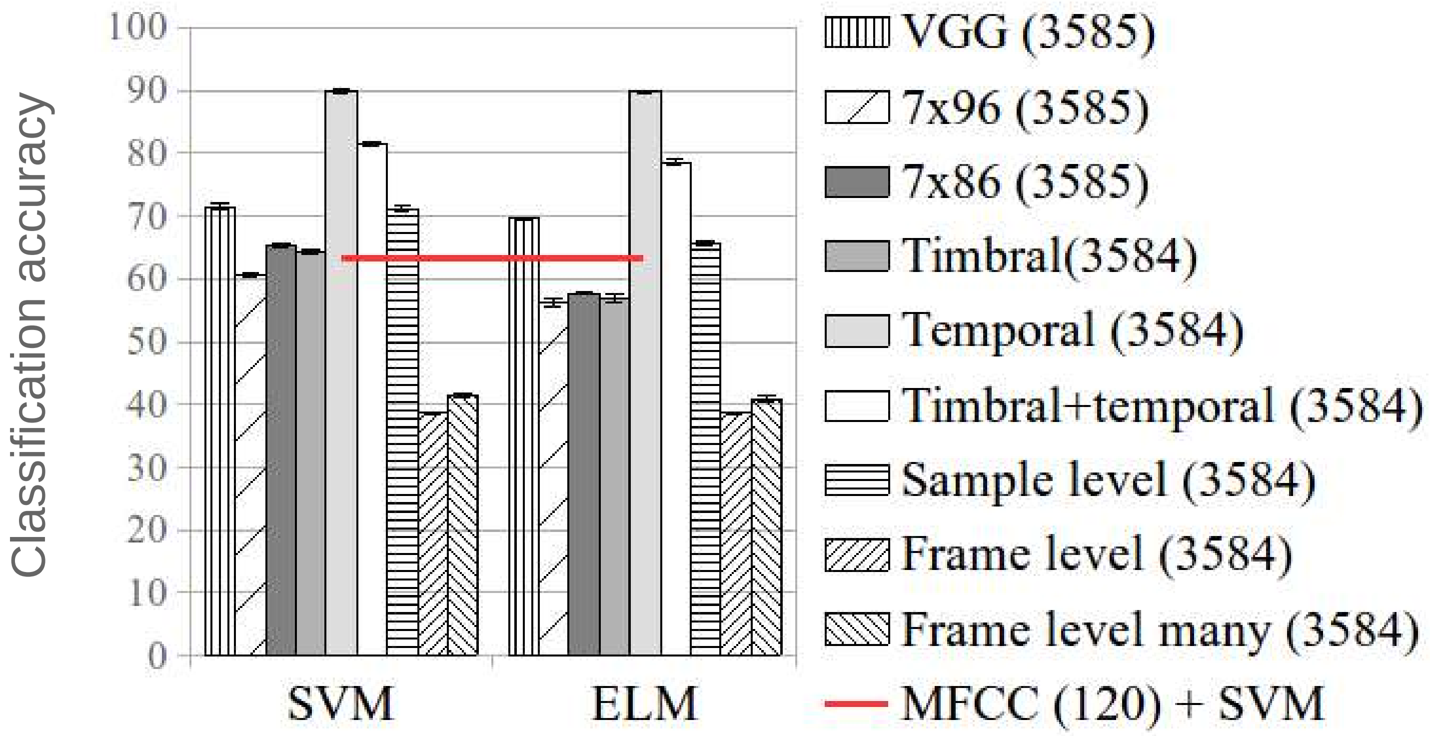
input	→	front-end	→	classifier	→	output
waveform log-mel spectrogram		*nine randomly weighted CNN architectures*		*SVM or ELM*		(music) audio tagging

Random CNN features: fault-filtered GTZAN



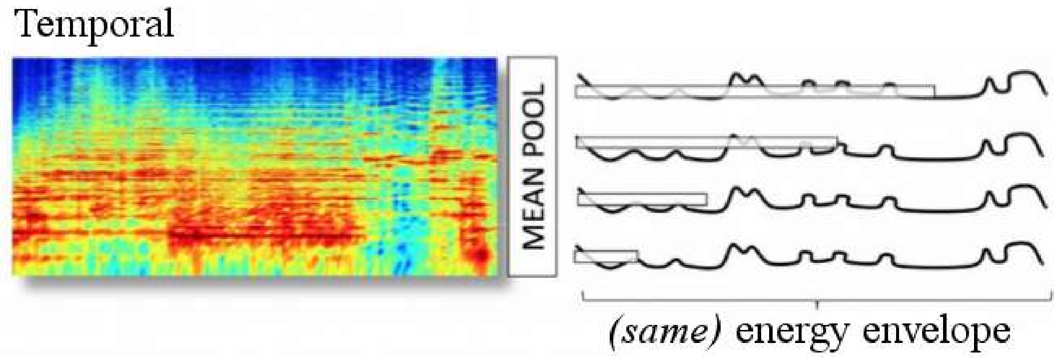
59.65 % (best random CNN) < 82.1 % (SOTA)

Random CNN features: Extended Ballroom

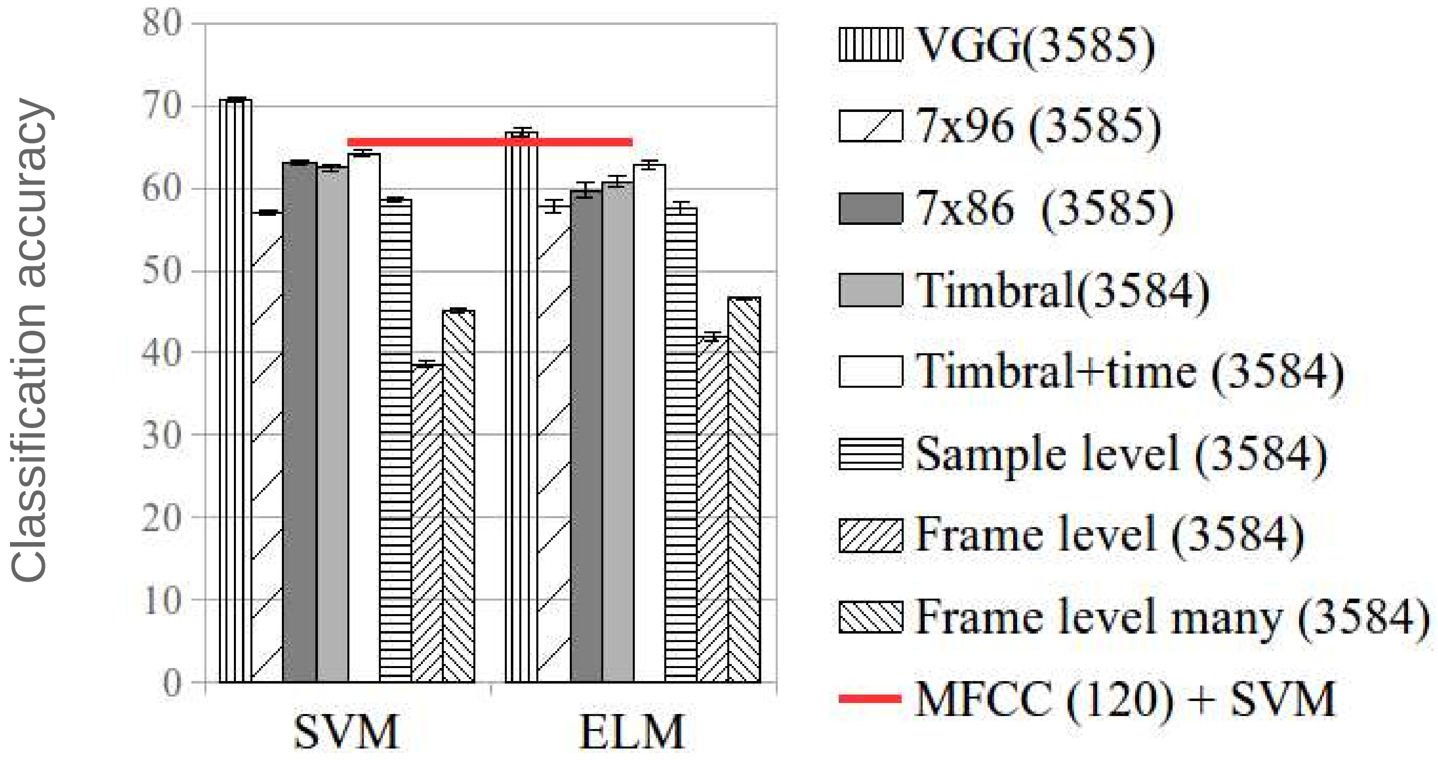


89.82 % (best random CNN) < 93.7 % (SOTA)

Do you remember the *temporal* CNN?



Random CNN features: Urban Sound 8k



70.74 % (best random CNN) < 73 % (SOTA)

Conclusions

- Main CNN front-ends for audio-classification where presented

- **Spectrogram front-ends > waveform front-ends**
- Waveform front-ends: **sample-level >> frame-level many > frame-level**
- Spectrogram front-ends: **7x86 > 7x96**

- One can achieve reasonable results without using domain knowledge
- Domain knowledge intuitions are valid guides for designing CNN-based models

Randomly weighted CNNs for audio classification: a personal (re)view

Jordi Pons

jordipons.me – @jordiponsdotme

Music Technology Group
Universitat Pompeu Fabra, Barcelona