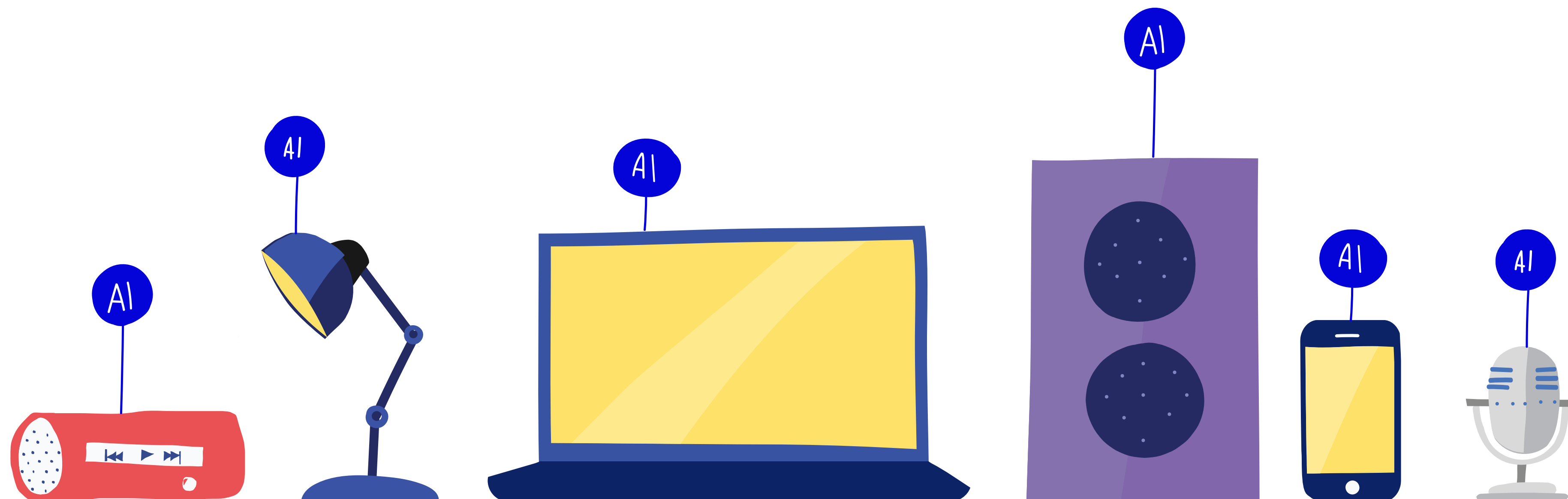




# neural networks on the edge

GDR BioComp 2019 — IRCICA — 2019-05-15  
Mathieu Poumeyrol — Principal Engineer — Snips  
@kalizoy

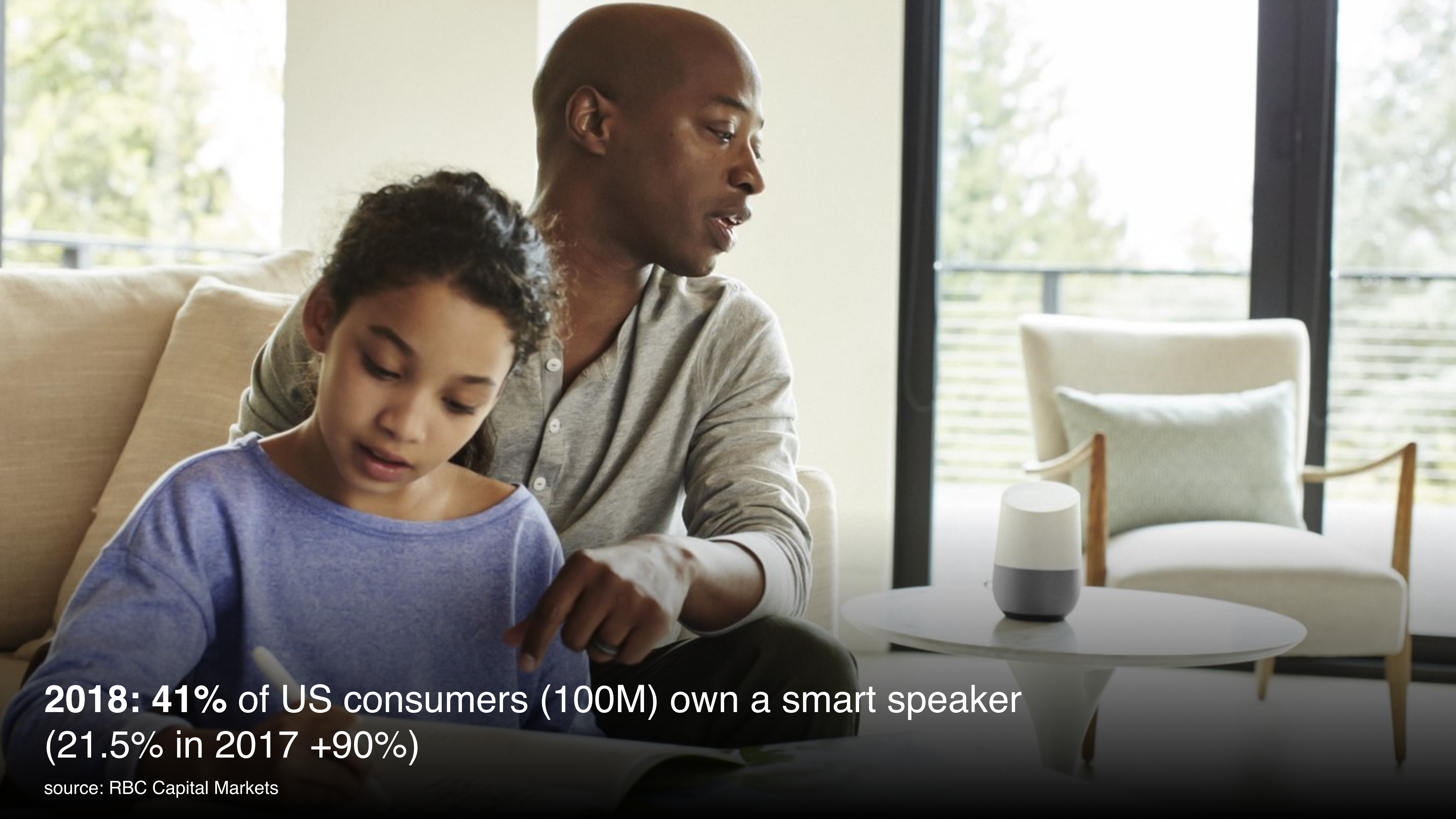


# Voice Interfaces

**Privacy concerns  
& Snips approach**







**2018: 41% of US consumers (100M) own a smart speaker  
(21.5% in 2017 +90%)**

source: RBC Capital Markets



INDY/TECH

# AMAZON ECHO SENDS LONG RECORDING OF COUPLE'S PRIVATE CONVERSATION TO RANDOM PERSON

The Amazon Echo, a voice-controlled virtual assistant, is seen at its product launch for Britain and Germany in London, Britain / REUTERS/Peter Hobson

'Unplug your Alexa devices right now. You're being hacked.'

c|net

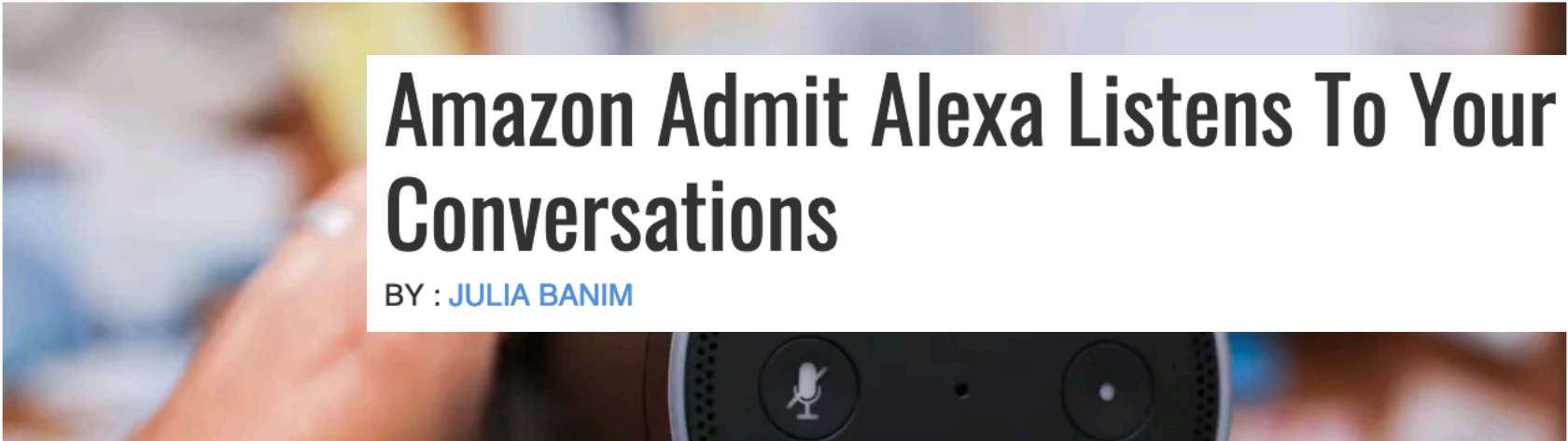
REVIEWS NEWS VIDEO HOW TO SMART HOME CARS DEALS DOWNLOAD

## Amazon Echo's calling feature includes a major privacy flaw

The only way to block someone right now is to turn the new feature off, a user discovers and reports.

TIM MOYNIHAN GEAR 12.05.16 9:00 AM

# ALEXA AND GOOGLE HOME RECORD WHAT YOU SAY. BUT WHAT HAPPENS TO THAT DATA?



## Amazon Admit Alexa Listens To Your Private Conversations

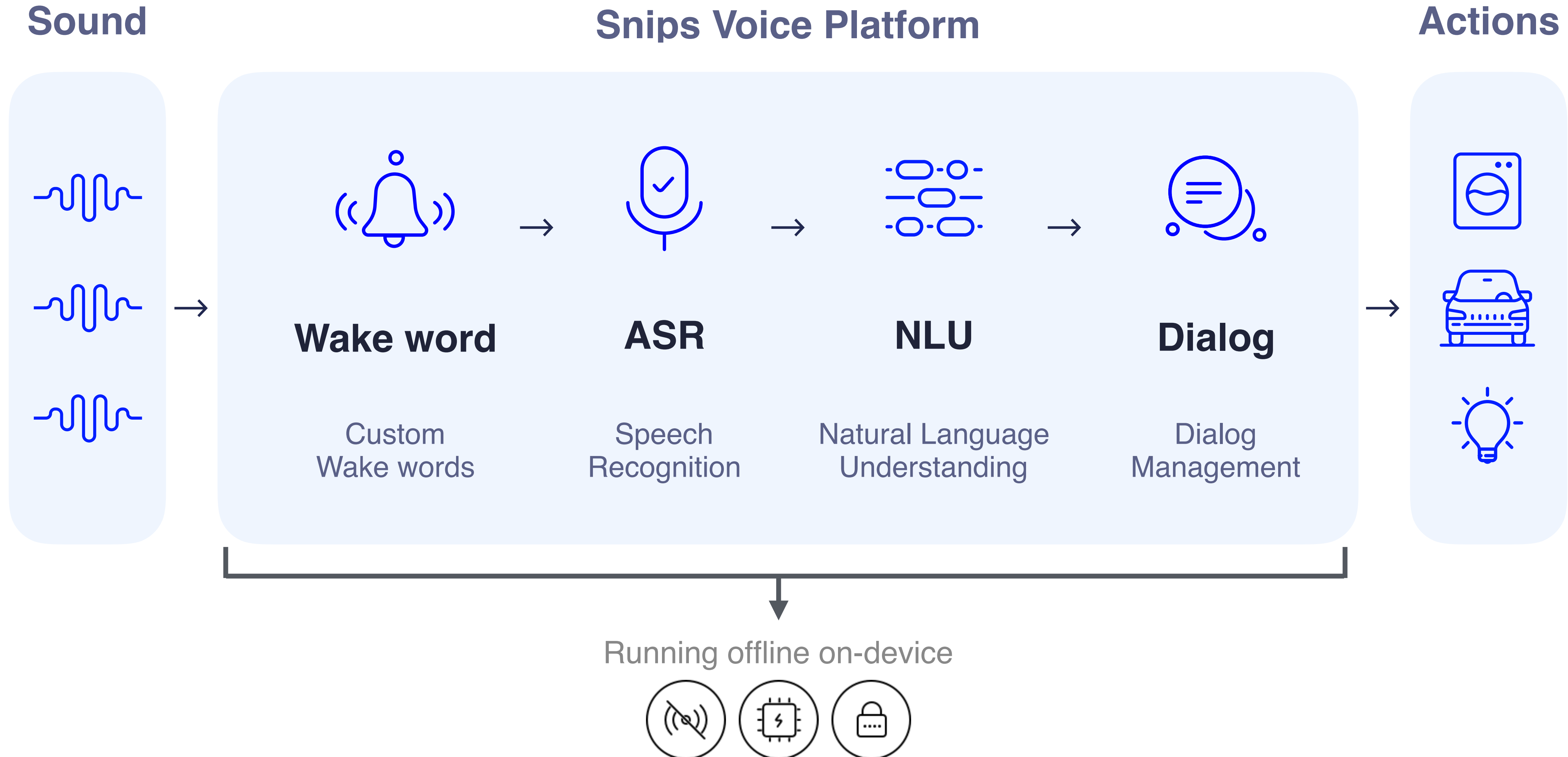
BY : JULIA BANIM

## Google Home Mini flaw left smart speaker recording everything

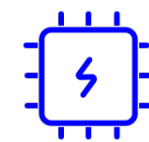
Updated: Google has released a firmware update to fix a Home Mini bug that made the device a privacy threat.

By  Liam Tung | October 11, 2017 -- 13:00 GMT (14:00 BST) | Topic: [Google](#)

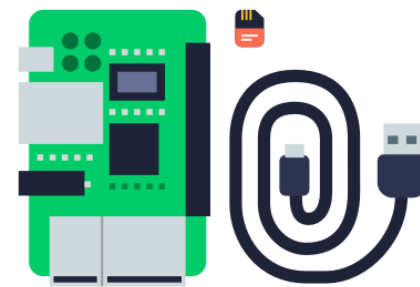
# Anatomy of a voice assistant



# Running on the edge



## Example Target Hardware



Raspberry Pi 3  
**1GB RAM**  
**1.4GHz 4xCPU**  
**ARM v8**  
**35\$**



**Wake word**



**Speaker identification**

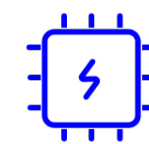


**Speech recognition**

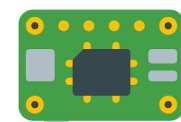
**Intent recognition and processing**

# Running on the edge... in multiple rooms

## Satellite



Example Target Hardware



Raspberry Pi 0  
**512MB RAM**  
**1GHz CPU**  
**ARM v6 + VFP**  
**5\$**

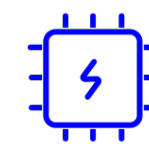


Wake word

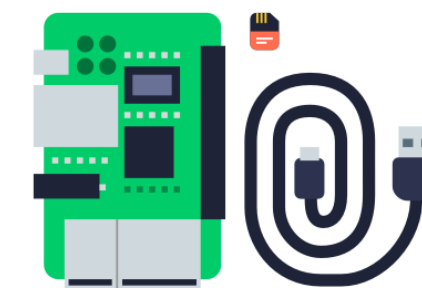


Speaker identification

Audio capture



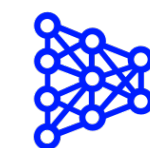
Example Target Hardware



Raspberry Pi 3  
**1GB RAM**  
**1.4GHz 4xCPU**  
**ARM v8**  
**35\$**



Wake word



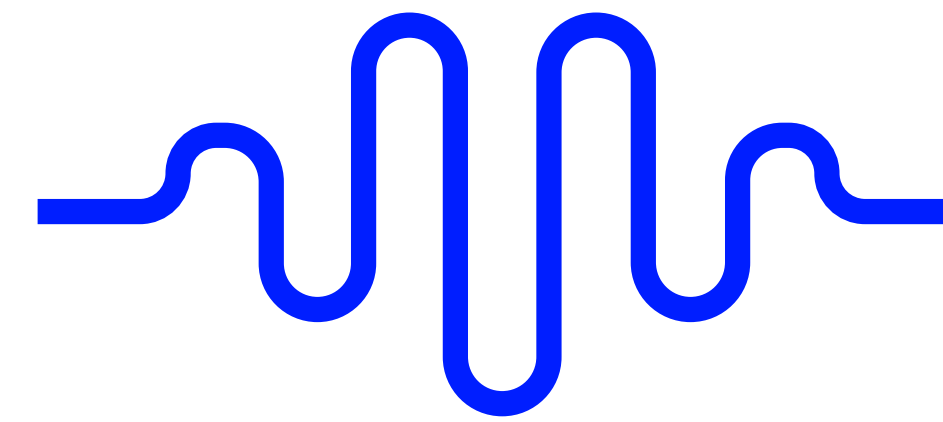
Speaker identification



Speech recognition

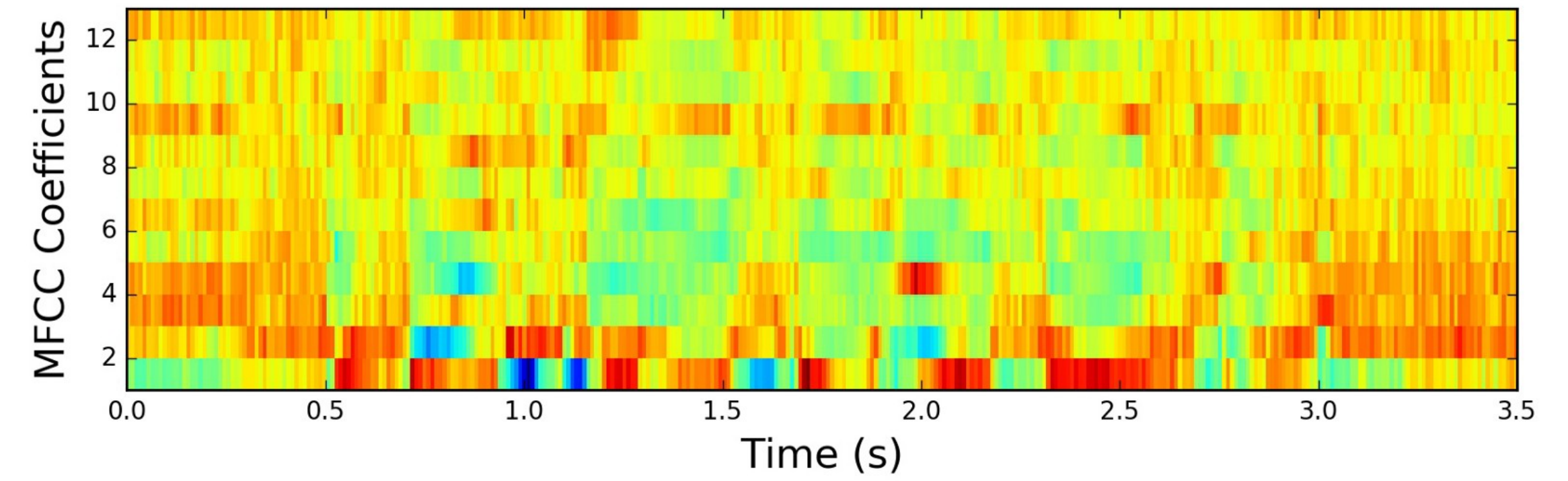
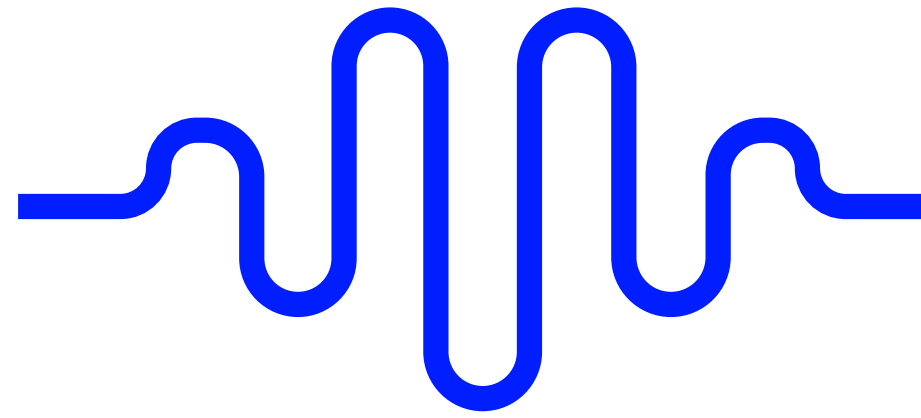
Intent recognition and processing

# Real-time voice and convolutions



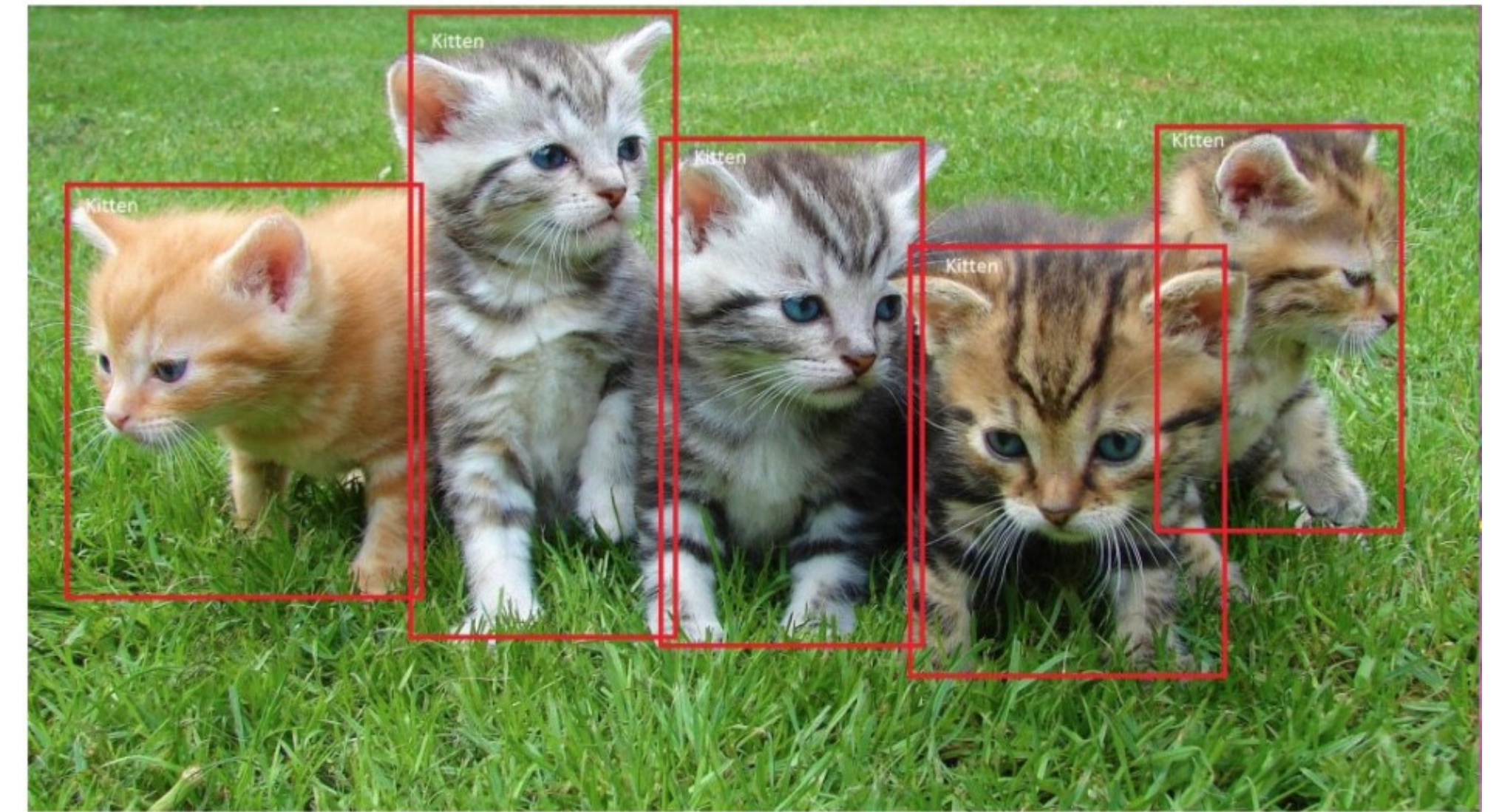
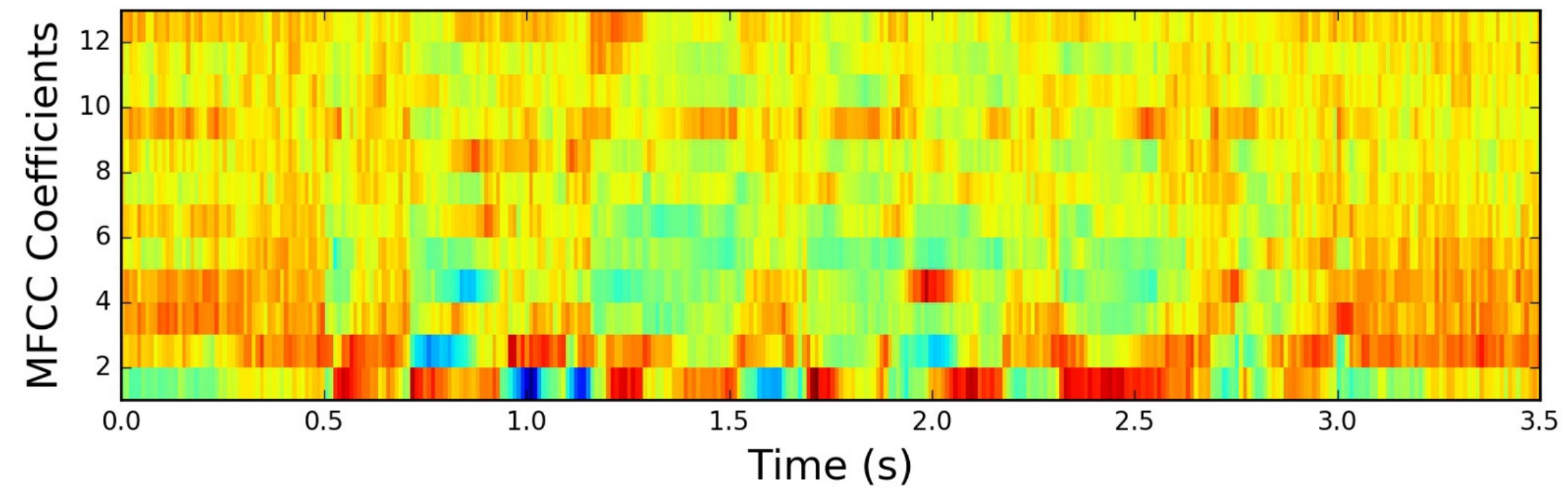


# Real time voice versus image



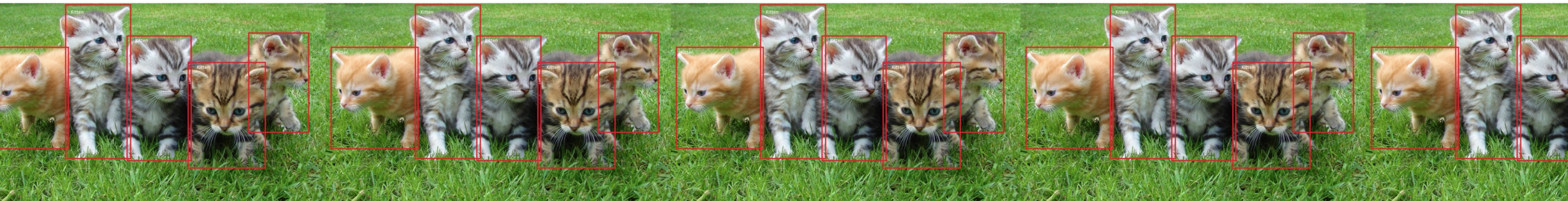


# Real time voice versus image





# Real time voice versus image





# Streaming

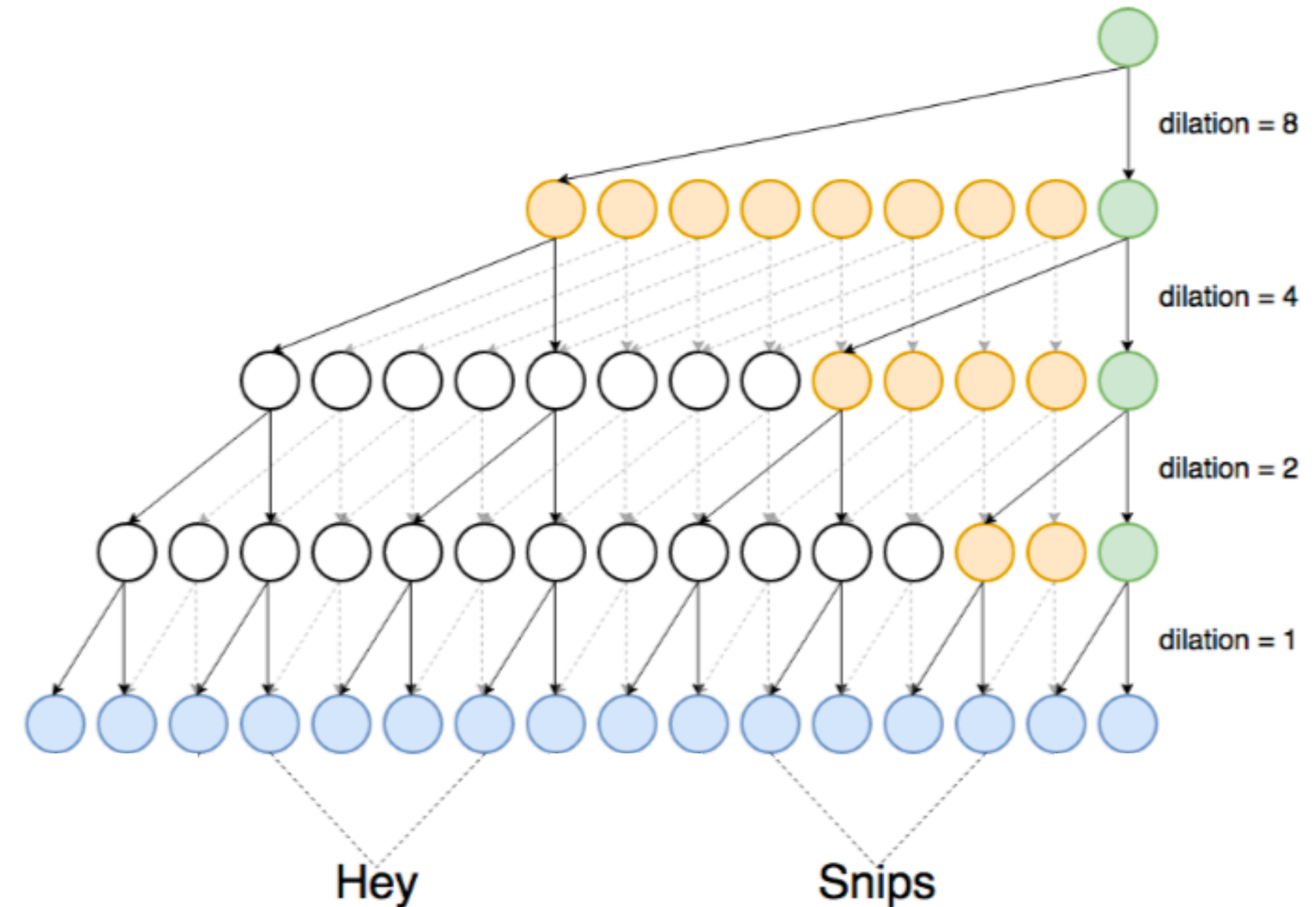
- Image Tensor: [width,height,depth] (299x299x3)
- Streaming audio: [time,feature] ( $\infty$ , 40)
- Streaming is a must because:
  - Infinite input wont work
  - and... you can't wait anyway.

# Recurring networks

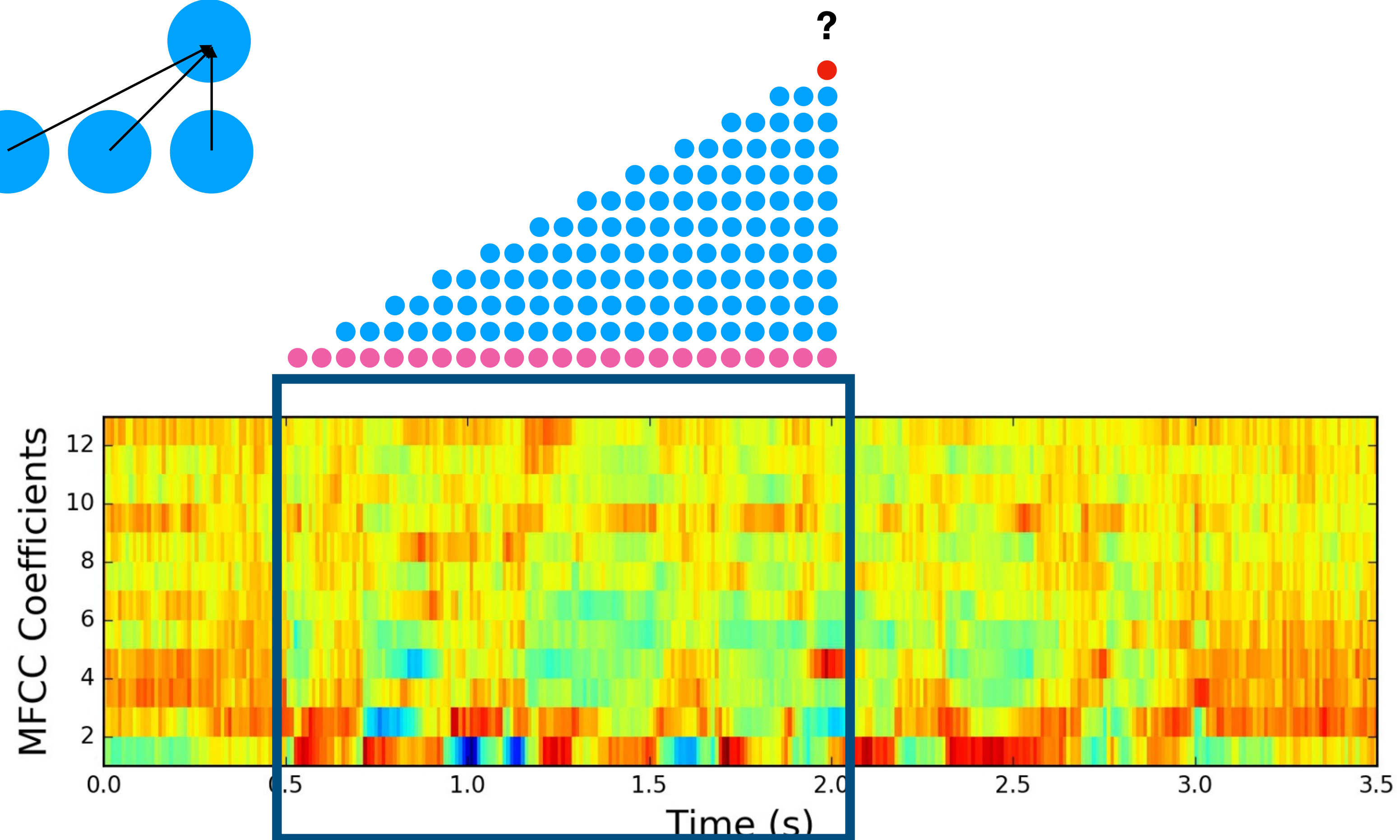
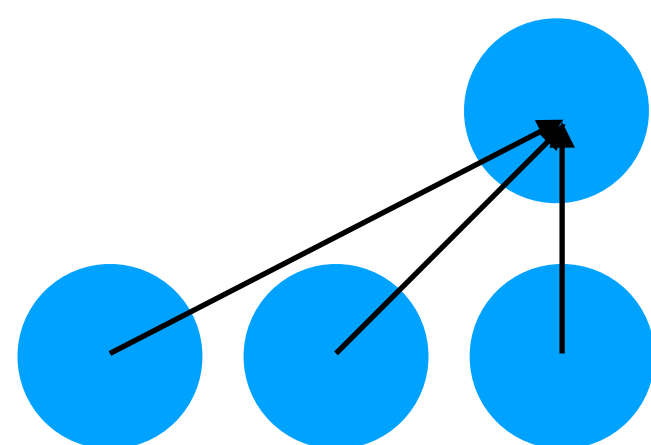
- Naturally streaming over chunks
- DeepSpeech
- State management issue with infinite inputs

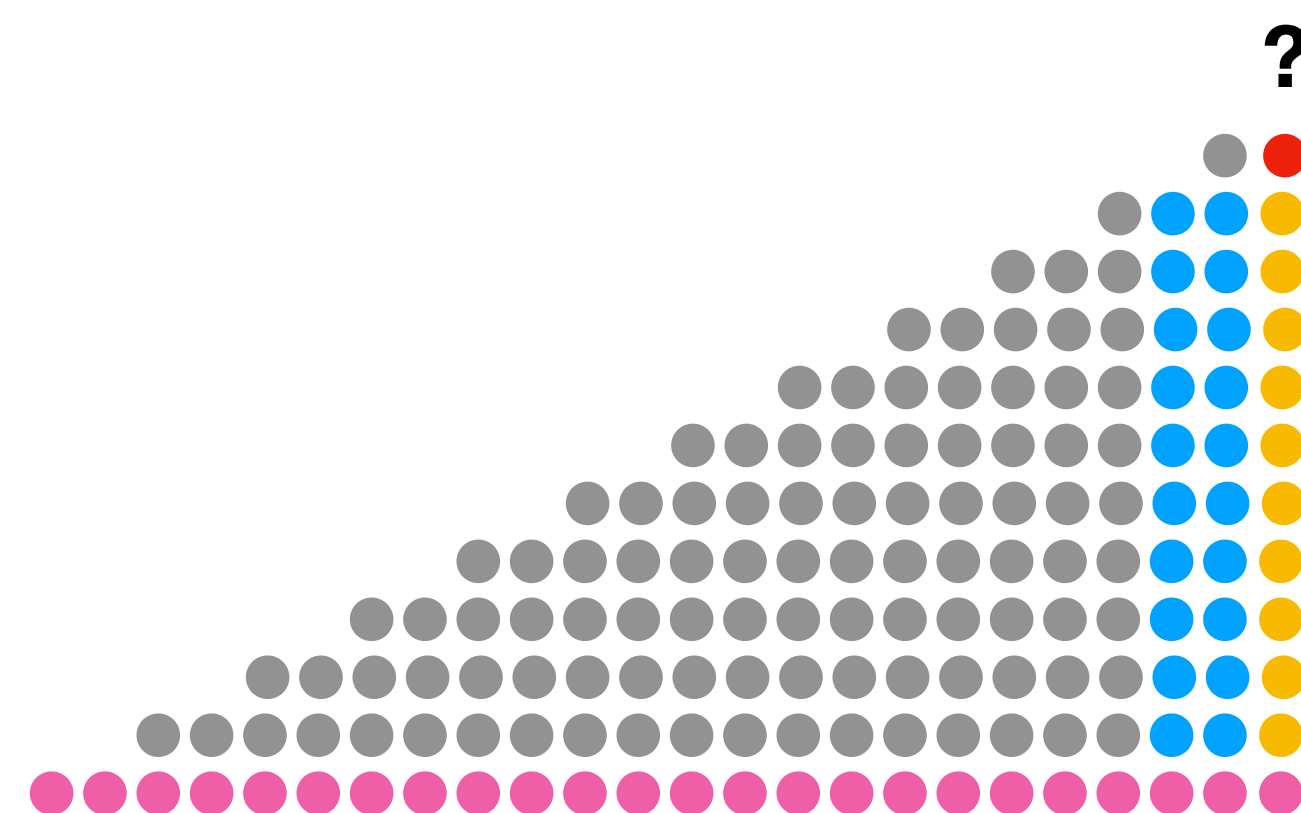
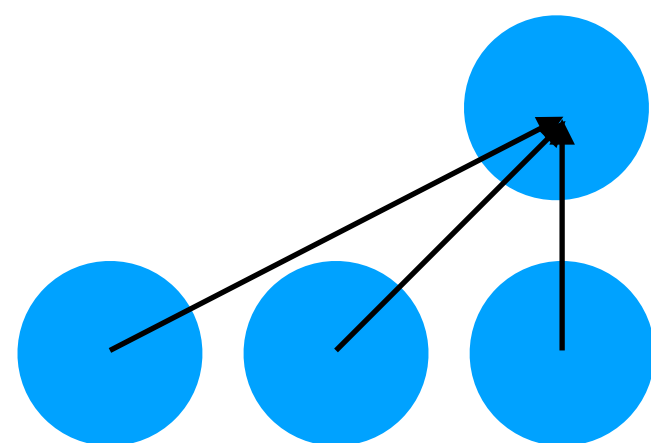
# WaveNet

- A network family developed for artificial voice generation
- Better results on the wake word task (paper, patent pending)
- Big stack of small convolutions:
  - wide receptive field: end-to-end
  - affordable: reusable values

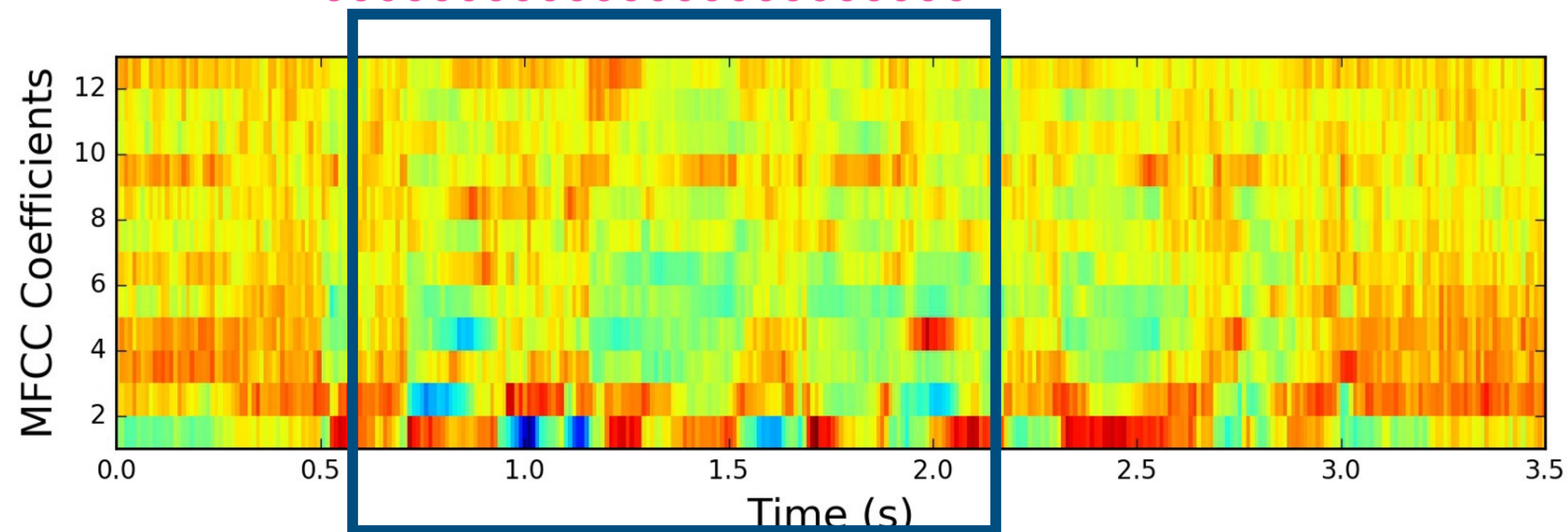








**With a small cache  
only ~4% of the values are new!**



# How to run it...

- Hardcode it
- Cajole TensorFlow (or Lite) into running it:
  - wire Variable2/Assign around the 48 convolution operators
- then run batches through the network



**Introducing...**



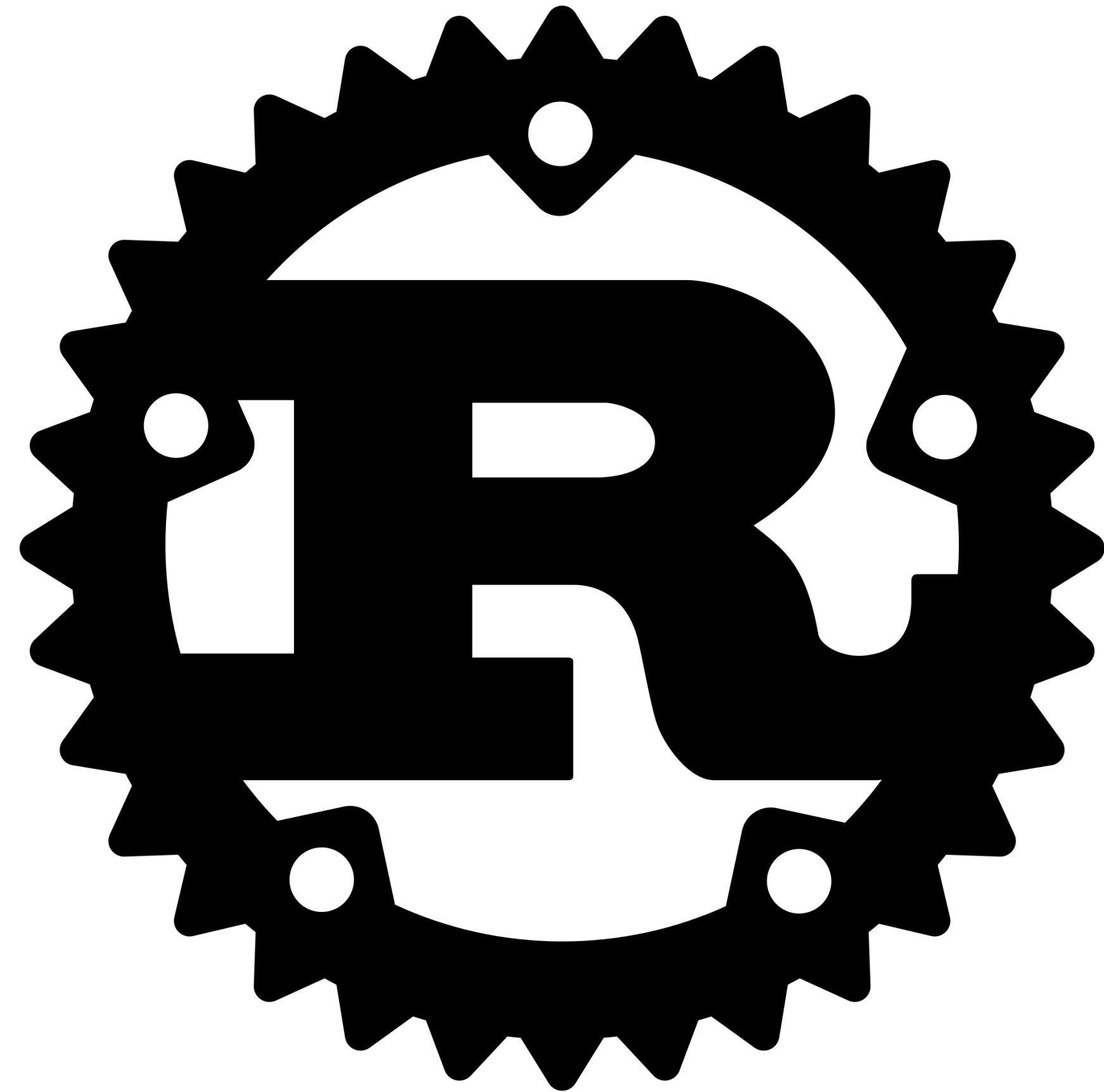
**tract**

# tract

- Snips Neural Network inference engine
- Small CPU friendly (ARMv6 to ARMv8)
- OpenSource
- In Rust
- Is not TensorFlow

# Tract is a Rust library

- Rust is Snips' go-to language for embedded software
- Portability
- Performance
- Safety





# TensorFlow support

- Read native TensorFlow format
- TensorFlow operator set is huge, and not specified
  - adding operators as needed
- More comprehensive than TensorFlow-Lite already



# ONNX support

- Read native ONNX format
- ONNX has a specified operator set, provides test suite
- tract supports about 85% of ONNX test suite

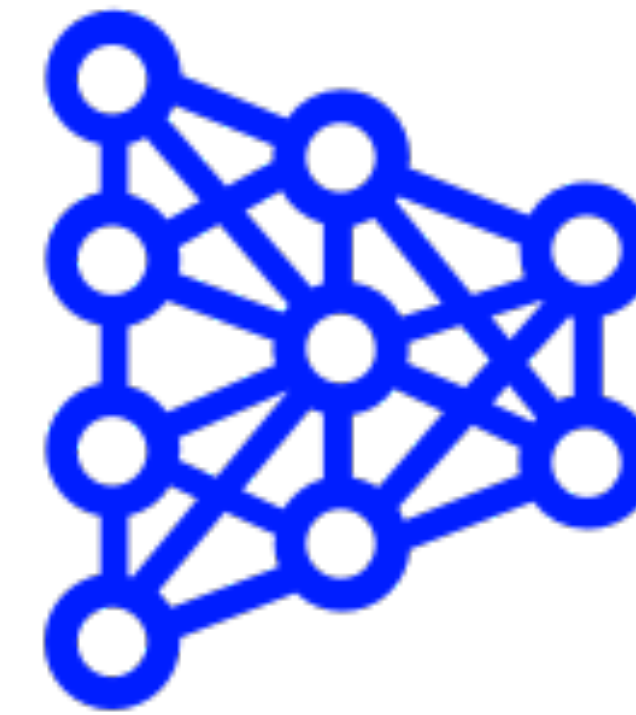


# tract is OpenSource

- tract is MIT/Apache-2
- no dependency under lawyer-scaring license



# **Neural Network Interpreter: Anatomy and issues**



# Minimum TensorFlow runner

- Parse TensorFlow protobuf format
- Inject input
- Cascade data through operators, repeat.

# Optimising interpreter

- Parse TensorFlow protobuf format
- **Type the network: tensors data types and shapes**
- **Declutter the network**
- **Propagate constants, reduce operator strength, ...**
- **“Codegen” logical operators to hardware ops**
- Inject inputs
- Run the operators



# Decluttering training networks

- WaveNet “Hey Snips”: 3080 ops -> 267 ops
- Inception v3: 1655 ops -> ~500 ops
- DeepSpeech: 114 ops -> 54 ops

```
0 Source input_lengths
1 Const lstm_fused_cell/Const
2 tf.Max lstm_fused_cell/Max
3 Cast lstm_fused_cell/ToInt64
4 Source input_node
5 Const transpose/perm
6 tf.Transpose transpose
7 Const Reshape/shape
8 tf.Reshape Reshape
9 Const h1
10 tf.Identity h1/read
11 Gemm MatMul
12 Const b1
13 tf.Identity b1/read
14 Add::Binary Add
15 Relu Relu
16 Const Minimum/y
17 Min::Binary Minimum
18 Const h2
19 tf.Identity h2/read
20 Gemm MatMul_1
21 Const b2
22 tf.Identity b2/read
23 Add::Binary Add_1
24 Relu Relu_1
25 Const Minimum_1/y
26 Min::Binary Minimum_1
27 Const h3
28 tf.Identity h3/read
29 Gemm MatMul_2
30 Const b3
31 tf.Identity b3/read
32 Add::Binary Add_2
33 Relu Relu_2
34 Const Minimum_2/y
35 Min::Binary Minimum_2
36 Const Reshape_1/shape
37 tf.Reshape Reshape_1
38 tf.VariableV2 previous_state_c
39 tf.Identity previous_state_c/read
40 tf.VariableV2 previous_state_h
41 tf.Identity previous_state_h/read
42 Const lstm_fused_cell/kernel
43 tf.Identity lstm_fused_cell/kernel/read
44 Const lstm_fused_cell/zeros/shape_as_tensor
45 Const lstm_fused_cell/zeros/Const
46 tf.Fill lstm_fused_cell/zeros
47 Const lstm_fused_cell/bias
48 tf.Identity lstm_fused_cell/bias/read
49 tf.BlockLSTM lstm_fused_cell/BlockLSTM
50 Const lstm_fused_cell/SequenceMask/Const_1
51 Const lstm_fused_cell/SequenceMask/Const
52 Const lstm_fused_cell/SequenceMask/Const_2
53 tf.Range lstm_fused_cell/SequenceMask/Range
54 Const lstm_fused_cell/SequenceMask/ExpandDims/dim
55 tf.ExpandDims lstm_fused_cell/SequenceMask/ExpandDims
56 Cast lstm_fused_cell/SequenceMask/Cast
57 Lesser::Binary lstm_fused_cell/SequenceMask/Less
58 Cast lstm_fused_cell/SequenceMask/Cast_1
59 Const lstm_fused_cell/transpose/perm
60 tf.Transpose lstm_fused_cell/transpose
61 Const lstm_fused_cell/ExpandDims/dim
62 tf.ExpandDims lstm_fused_cell/ExpandDims
63 Const lstm_fused_cell/Tile/multiples
64 Tile lstm_fused_cell/Tile
65 Mul::Binary lstm_fused_cell/mul
66 Const Reshape_2/shape
67 tf.Reshape Reshape_2
68 Const h5
69 tf.Identity h5/read
70 Gemm MatMul_3
71 Const b5
72 tf.Identity b5/read
73 Add::Binary Add_3
74 Relu Relu_3
75 Const Minimum_3/y
76 Min::Binary Minimum_3
77 Const h6
78 tf.Identity h6/read
79 Gemm MatMul_4
80 Const b6
81 tf.Identity b6/read
82 Add::Binary Add_4
83 Const raw_logits/shape
84 tf.Reshape raw_logits
85 LayerSoftmax Softmax
86 Const lstm_fused_cell/ExpandDims_1/dim
87 tf.ExpandDims lstm_fused_cell/ExpandDims_1
88 Const lstm_fused_cell/concat/axis
89 tf.ConvatV2 lstm_fused_cell/concat
90 Const lstm_fused_cell/range/start
91 Const lstm_fused_cell/range/limit
92 Const lstm_fused_cell/range/delta
93 tf.Range lstm_fused_cell/range
94 tf.Pack lstm_fused_cell/stack
95 tf.GatherNd lstm_fused_cell/GatherNd
96 tf.Assign Assign_2
97 Const lstm_fused_cell/ExpandDims_2/dim
98 tf.ExpandDims lstm_fused_cell/ExpandDims_2
99 Const lstm_fused_cell/concat_1/axis
100 tf.ConvatV2 lstm_fused_cell/concat_1
101 Const lstm_fused_cell/range_1/start
102 Const lstm_fused_cell/range_1/limit
103 Const lstm_fused_cell/range_1/delta
104 tf.Range lstm_fused_cell/range_1
105 tf.Pack lstm_fused_cell/stack_1
106 tf.GatherNd lstm_fused_cell/GatherNd_1
107 tf.Assign Assign_3
108 tf.Identity logits
109 Const zeros/shape_as_tensor
110 Const zeros/Const
111 tf.Fill zeros
112 tf.Assign Assign
113 tf.Assign Assign_1
114 tf.Noop initialize_state
```



```
0 Const Const-117
1 Source input_node
2 PermuteAxes transpose
3 Const Reshape/shape
4 tf.Reshape Reshape
5 GemmUnaryA MatMul
6 Add::UnaryA Add
7 Relu Relu
8 Min::UnaryA Minimum
9 GemmUnaryA MatMul_1
10 Add::UnaryA Add_1
11 Relu Relu_1
12 Min::UnaryA Minimum_1
13 GemmUnaryA MatMul_2
14 Add::UnaryA Add_2
15 Relu Relu_2
16 Min::UnaryA Minimum_2
17 Const Reshape_1/shape
18 tf.Reshape Reshape_1
19 tf.VariableV2 previous_state_c
20 tf.VariableV2 previous_state_h
21 Const Const-118
22 Const Const-119
23 Const Const-120
24 Const Const-121
25 Const Const-122
26 tf.BlockLSTM lstm_fused_cell/BlockLSTM
27 Mul::UnaryA lstm_fused_cell/mul
28 Const Reshape_2/shape
29 tf.Reshape Reshape_2
30 GemmUnaryA MatMul_3
31 Add::UnaryA Add_3
32 Relu Relu_3
33 Min::UnaryA Minimum_3
34 GemmUnaryA MatMul_4
35 Add::UnaryA Add_4
36 Const raw_logits/shape
37 tf.Reshape raw_logits
38 LayerSoftmax Softmax
39 AddDims lstm_fused_cell/ExpandDims_1
40 Const lstm_fused_cell/concat/axis
41 tf.ConvatV2 lstm_fused_cell/concat
42 Const Const-129
43 tf.GatherNd lstm_fused_cell/GatherNd
44 tf.Assign Assign_2
45 AddDims lstm_fused_cell/ExpandDims_2
46 Const lstm_fused_cell/concat_1/axis
47 tf.ConvatV2 lstm_fused_cell/concat_1
48 Const Const-135
49 tf.GatherNd lstm_fused_cell/GatherNd_1
50 tf.Assign Assign_3
51 tf.Identity logits
52 Const Const-136
53 tf.Assign Assign
54 Const Const-134
55 tf.Assign Assign_1
56 tf.Noop initialize_state
```

Fully connected 1  
Fully connected 2  
Fully connected 3

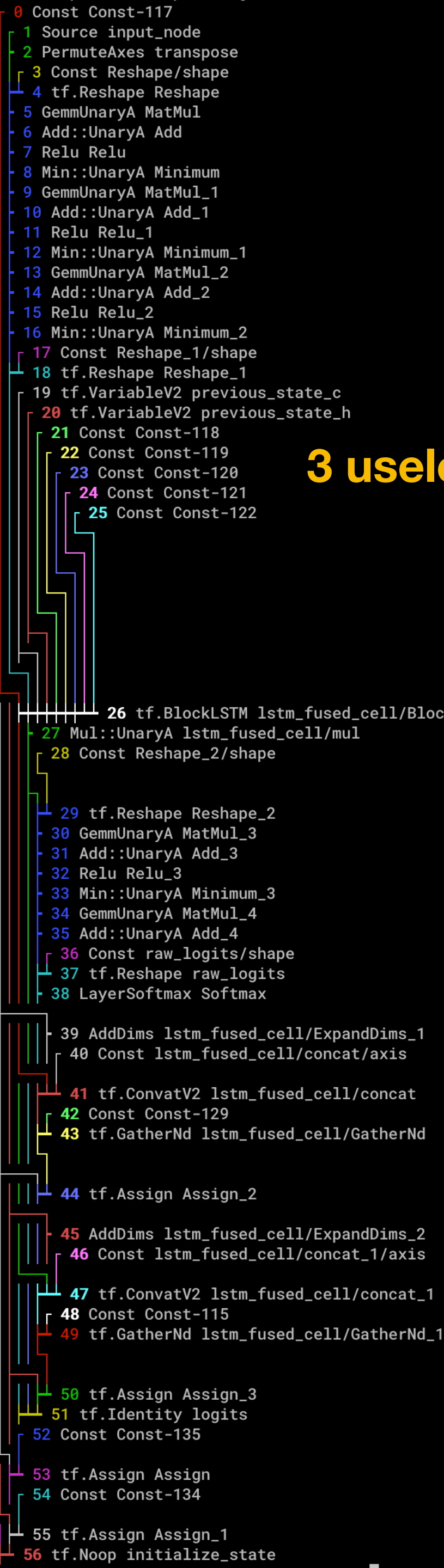
3 useless input parameters (all zeros)

LSTM Layer

5 out of 7 outputs dropped

Fully connected 4  
Fully connected 5

LSTM state wiring



- 5 GemmUnaryA MatMul
- 6 Add::UnaryA Add
- 7 Relu Relu
- 8 Min::UnaryA Minimum
- 9 GemmUnaryA MatMul\_1
- 10 Add::UnaryA Add\_1
- 11 Relu Relu\_1
- 12 Min::UnaryA Minimum\_1
- 13 GemmUnaryA MatMul\_2
- 14 Add::UnaryA Add\_2
- 15 Relu Relu\_2
- 16 Min::UnaryA Minimum\_2



# Bigger semantic transforms: Streaming

- Load, analyse, type original network
  - **implies reasoning on the “streaming” dimension**
- **Translate operators to manipulate “pulses” of data:**
  - **most operators are actually unchanged**
  - **some need addition of a stateful op (e.g. conv on time axis)**
  - **some are impossible to translate (e.g. softmax on time axis)**
- Produces a **stateful** network that operates over pulses

## **Challenge: Highly immature field**

- Deep Learning revolution circa 2010
- TensorFlow introduced end of 2015
- Example: dilated convolutions re-discovered in 2016

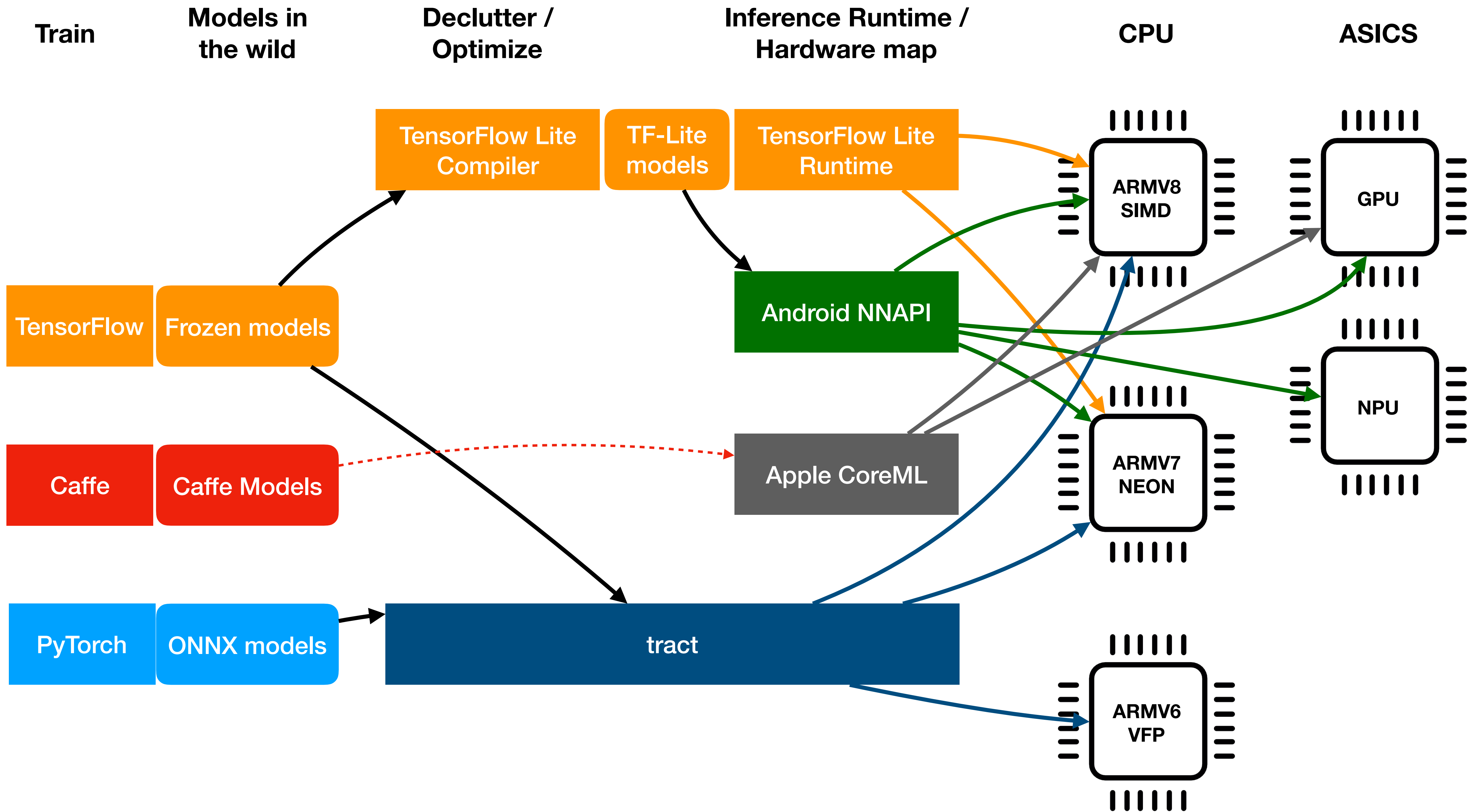
# Challenge: Frameworks are designed for training

- Most important contributors: Google, Facebook.
- Inference on cloud is comparatively easy.
- Mood may be changing: new Google Mobile ASR
- Available models are training oriented (declutter)



# Challenge: Inference side is fragmented

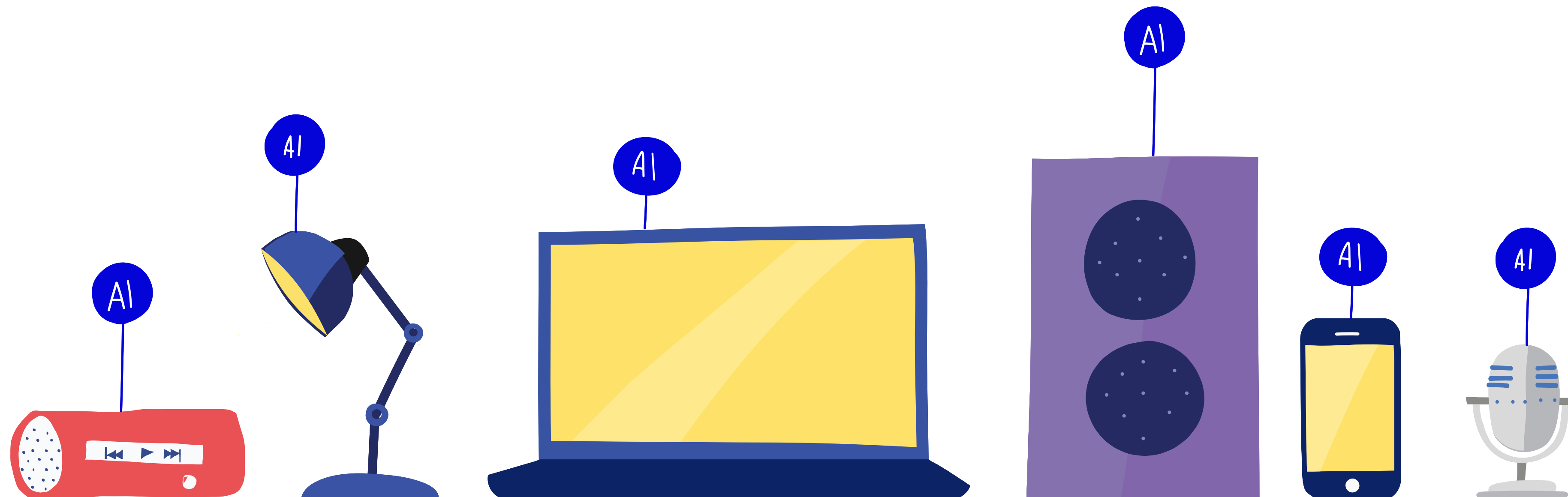
- Google: Android NNAPI, ~~TensorFlow Mobile~~, TensorFlow Lite
- Microsoft ONNX Runtime
- Apple CoreML and BNNS
- ARM NN SDK
- tract
- but also: TVM, TensorRT, NCNN





[github.com/snipsco/tract](https://github.com/snipsco/tract)

GDR BioComp 2019 — IRCICA — 2019-05-15  
Mathieu Poumeyrol — Principal Engineer — Snips  
@kalizoy

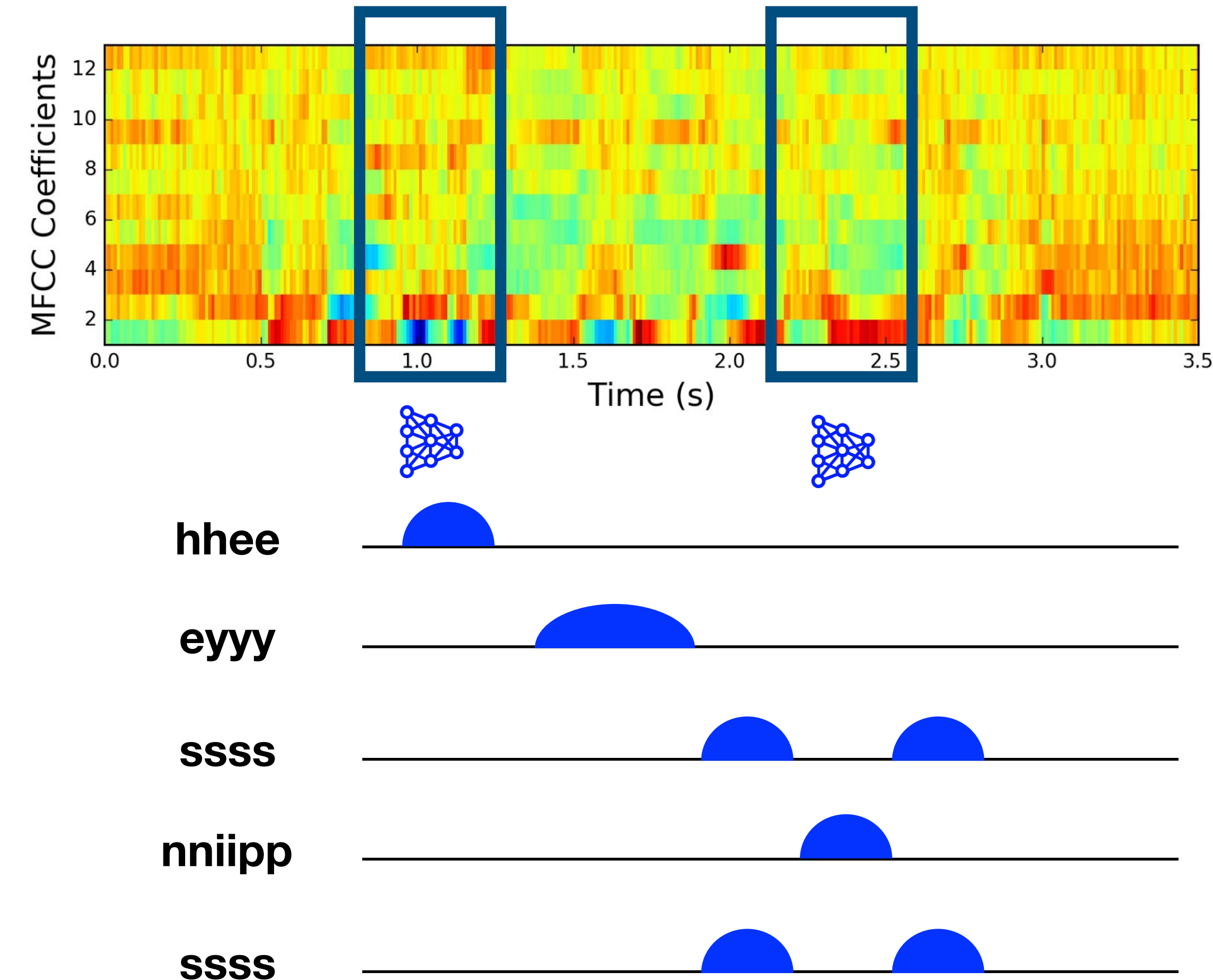




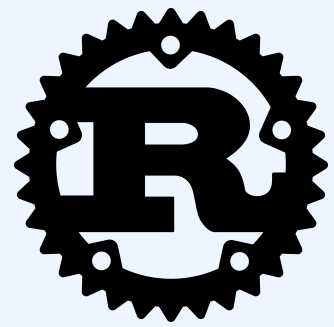
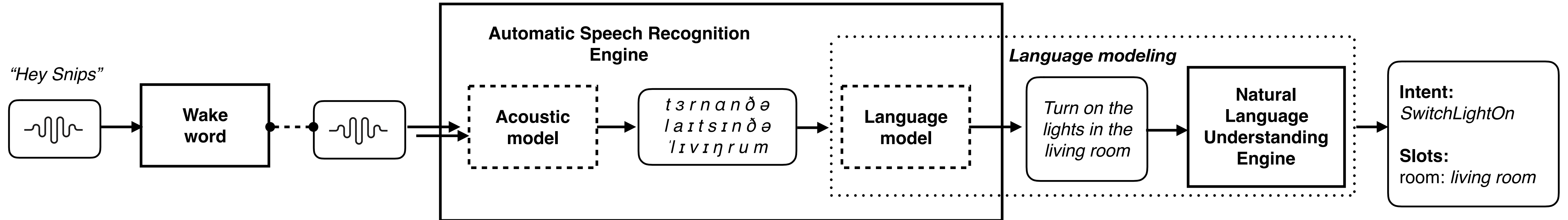


# Class-based

hhhheeeeyyyy sssnniiippii ssss



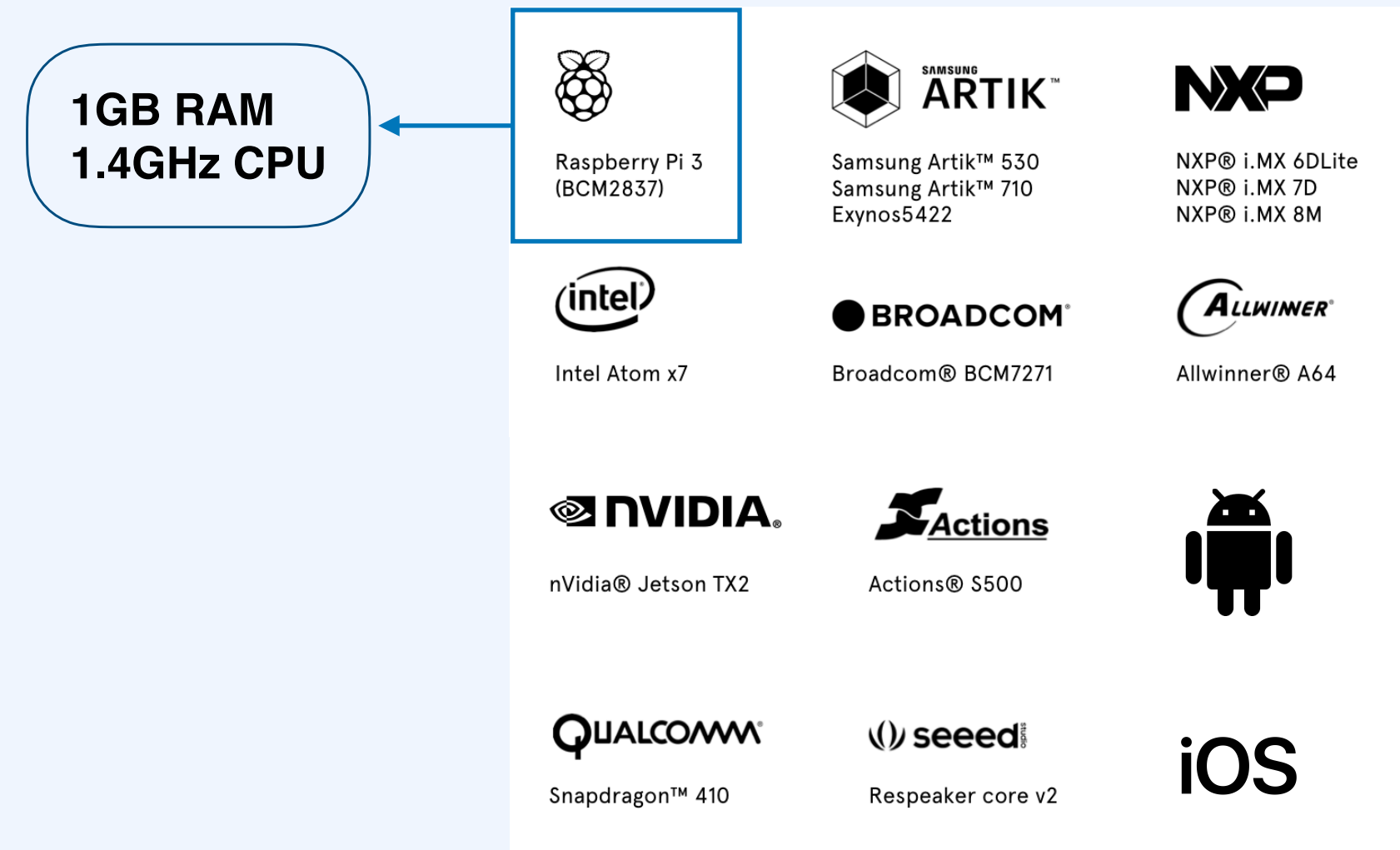
- Detects phones (“phonèmes” in french)
- Aggregate phone detections over 2 seconds to make a wake word decision
- Require “aligned” training data



## Rust language for inference

- **Portability** - write code once and cross-compile to any modern hardware architecture.
- **Performance** - high-level features without runtime performance penalty.
- **Safety** - rustc compiler makes it difficult to put memory-unsafe code into a production environment.

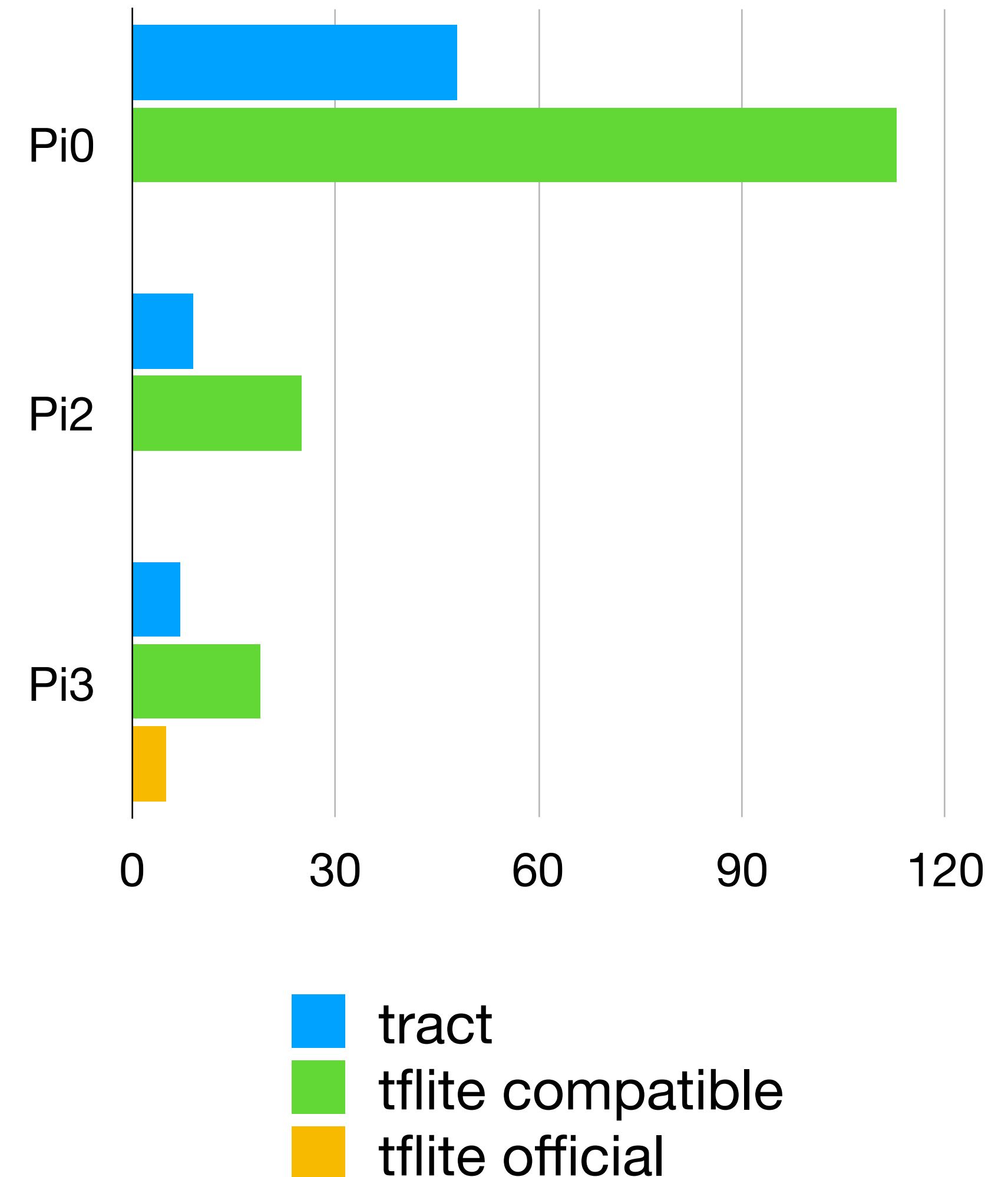
## Tested and certified to run on



# Performance on Inception v3

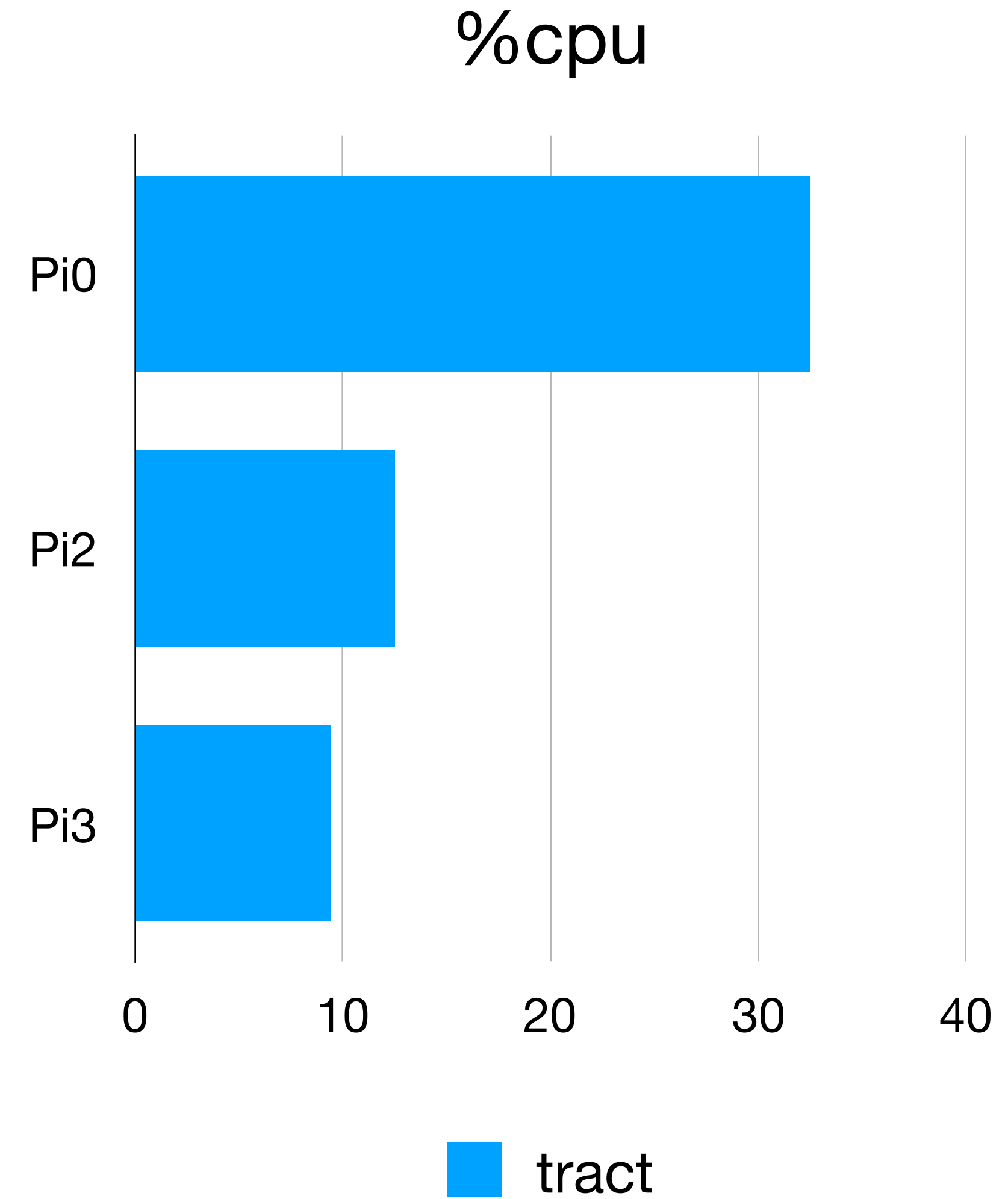
(seconds)

- TensorFlow hello world
- ImageNet challenge
- 299x299 images -> 1000 categories
- Against TensorFlow-lite
- Official build runs on Raspi3 only

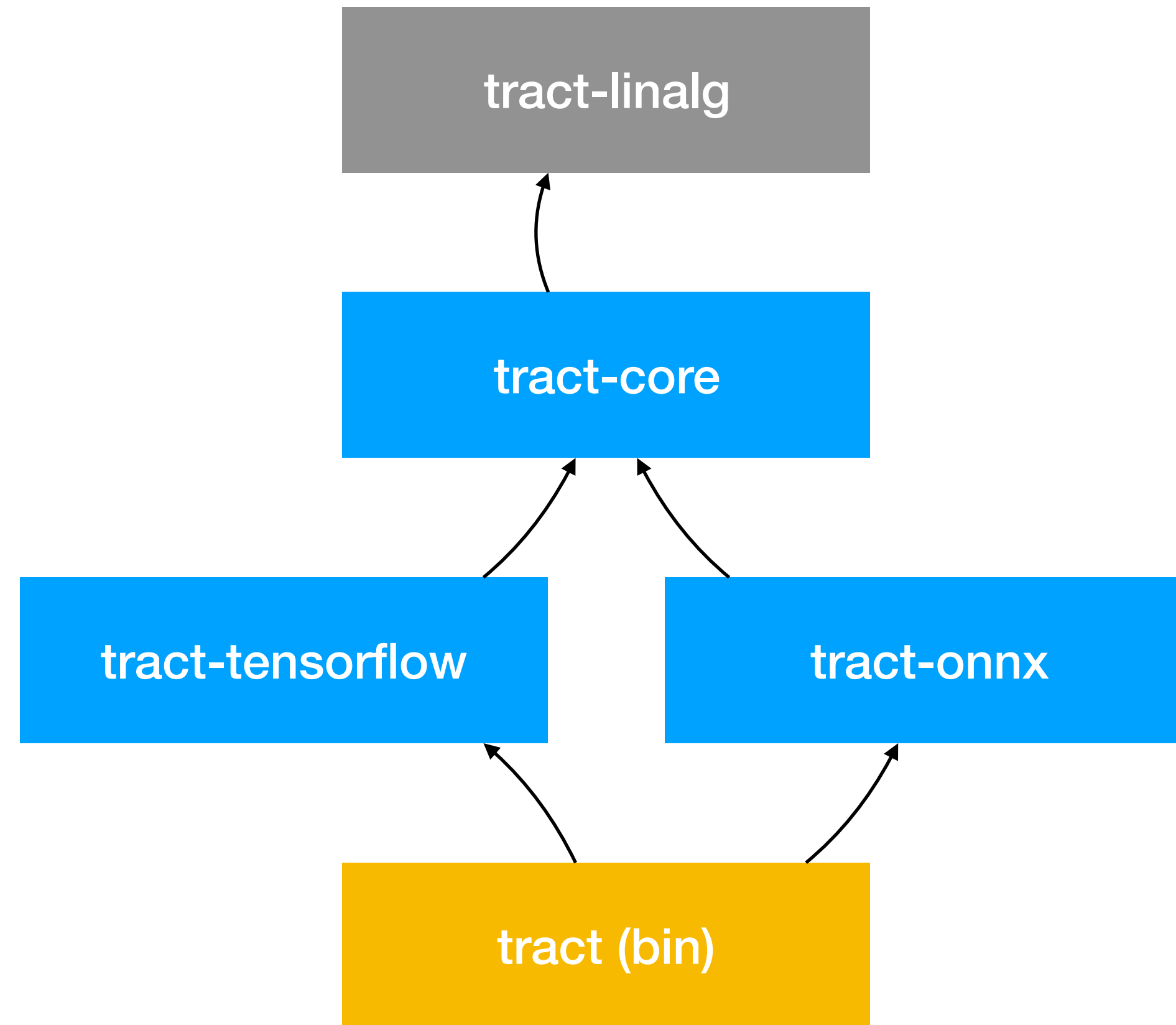


# Performance on Snips Wake word (WaveNet)

- one feature every 10ms
- 8-wide pulse: one pass every 80ms
- no TensorFlow-Lite comparison...
- no “pulse” mode
- missing operators and modes (tanh, addn, 1d dilated convolution, ...)







```

use tract_core::prelude::*;
use tract_core::ndarray;
use tract_core::datum::Datum;

fn main() -> TractResult<()> {
    let mut model = tract_tensorflow::tensorflow().model_for_path("mobilenet_v2_1.4_224_frozen.pb")?;
    model.set_input_fact(0, TensorFact::dt_shape(f32::datum_type(), tvec!(1, 224, 224, 3)))?;
    let model = model.into_optimized()?;
    let plan = SimplePlan::new(&model)?;

    let image = image::open("grace_hopper.jpg").unwrap().to_rgb();
    let resized = image::imageops::resize(&image, 224, 224, ::image::FilterType::Triangle);
    let image:Tensor = ndarray::Array4::from_shape_fn((1, 224, 224, 3), |(_, y, x, c)| {
        resized[(x as _, y as _)][c] as f32 / 255.0
    }).into();

    let result = plan.run(tvec!(image))?;
    let best = result[0].to_array_view:::<f32>()?.iter().cloned().enumerate().max_by(|a,b| a.1.partial_cmp(&b.1).unwrap());
    println!("result: {:?}", best);
    Ok(())
}

```

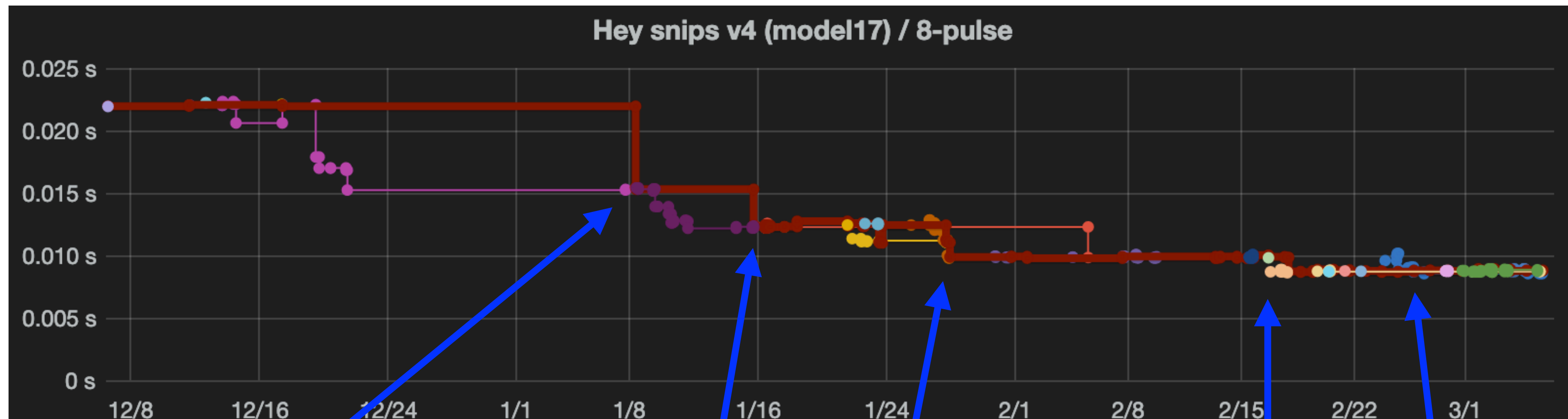
+-----+		+-----+	
5	Operation: <b>Conv::DepthWise&lt;F32&gt;</b>	Name: MobilenetV2/expanded_conv/depthwise/depthwise	
+-----+		+-----+	
	Real: <b>9.600 ms/i 12%</b> User: <b>9.600 ms/i 12%</b> Sys: <b>0.000 ms/i 0%</b>		
+-----+		+-----+	
	Input 0: Node #4/0 1x112x112x48xF32		
+-----+		+-----+	
	Output 0: 1x112x112x48xF32		
+-----+		+-----+	
	Attr padding: s: "SAME"		
	Attr dilations: list {i: 1 i: 1 i: 1 i: 1}		
	Attr strides: list {i: 1 i: 1 i: 1 i: 1}		
	Attr T: type: DT_FLOAT		
	Attr data_format: s: "NHWC"		
+-----+		+-----+	
+-----+		+-----+	
17	Operation: <b>Conv::DepthWise&lt;F32&gt;</b>	Name: MobilenetV2/expanded_conv_1/depthwise/depthwise	
+-----+		+-----+	
	Real: <b>7.232 ms/i 9%</b> User: <b>7.232 ms/i 9%</b> Sys: <b>0.000 ms/i 0%</b>		
+-----+		+-----+	
	Input 0: Node #16/0 1x112x112x144xF32		
+-----+		+-----+	
	Output 0: 1x56x56x144xF32		
+-----+		+-----+	
	Attr strides: list {i: 1 i: 2 i: 2 i: 1}		
	Attr padding: s: "SAME"		
	Attr T: type: DT_FLOAT		
	Attr dilations: list {i: 1 i: 1 i: 1 i: 1}		
	Attr data_format: s: "NHWC"		
+-----+		+-----+	
+-----+		+-----+	
29	Operation: <b>Conv::DepthWise&lt;F32&gt;</b>	Name: MobilenetV2/expanded_conv_2/depthwise/depthwise	
+-----+		+-----+	
	Real: <b>10.054 ms/i 12%</b> User: <b>10.054 ms/i 12%</b> Sys: <b>0.000 ms/i 0%</b>		
+-----+		+-----+	
	Input 0: Node #28/0 1x56x56x192xF32		
+-----+		+-----+	
	Output 0: 1x56x56x192xF32		
+-----+		+-----+	
	Attr T: type: DT_FLOAT		
	Attr padding: s: "SAME"		
	Attr dilations: list {i: 1 i: 1 i: 1 i: 1}		
	Attr strides: list {i: 1 i: 1 i: 1 i: 1}		
	Attr data_format: s: "NHWC"		
+-----+		+-----+	
+-----+		+-----+	
54	Operation: <b>Conv::DepthWise&lt;F32&gt;</b>	Name: MobilenetV2/expanded_conv_4/depthwise/depthwise	
+-----+		+-----+	
	Real: <b>4.167 ms/i 5%</b> User: <b>4.166 ms/i 5%</b> Sys: <b>0.000 ms/i 0%</b>		
+-----+		+-----+	
	Input 0: Node #53/0 1x28x28x288xF32		
+-----+		+-----+	
	Output 0: 1x28x28x288xF32		
+-----+		+-----+	
	Attr data_format: s: "NHWC"		
	Attr dilations: list {i: 1 i: 1 i: 1 i: 1}		
	Attr strides: list {i: 1 i: 1 i: 1 i: 1}		
	Attr T: type: DT_FLOAT		
	Attr padding: s: "SAME"		
+-----+		+-----+	
+-----+		+-----+	
67	Operation: <b>Conv::DepthWise&lt;F32&gt;</b>	Name: MobilenetV2/expanded_conv_5/depthwise/depthwise	
+-----+		+-----+	
	Real: <b>4.368 ms/i 5%</b> User: <b>4.367 ms/i 5%</b> Sys: <b>0.000 ms/i 0%</b>		
+-----+		+-----+	
	Input 0: Node #66/0 1x28x28x288xF32		
+-----+		+-----+	
	Output 0: 1x28x28x288xF32		
+-----+		+-----+	
	Attr strides: list {i: 1 i: 1 i: 1 i: 1}		
	Attr T: type: DT_FLOAT		
	Attr padding: s: "SAME"		
	Attr data_format: s: "NHWC"		
	Attr dilations: list {i: 1 i: 1 i: 1 i: 1}		
+-----+		+-----+	

Most time consuming operations:  
**Conv::DepthWise<F32>** 17 calls: Real: 61.049 ms/i 73% User: 61.048 ms/i 73% Sys: 0.000 ms/i 0%  
**MatMulUnaryImplASimpleB** 35 calls: Real: 11.925 ms/i 14% User: 11.925 ms/i 14% Sys: 0.000 ms/i 0%  
**Add::UnaryA** 36 calls: Real: 3.455 ms/i 4% User: 3.453 ms/i 4% Sys: 0.001 ms/i 87%  
**tf.FusedBatchNorm** 17 calls: Real: 3.286 ms/i 4% User: 3.285 ms/i 4% Sys: 0.000 ms/i 4%  
**Relu6** 35 calls: Real: 2.592 ms/i 3% User: 2.592 ms/i 3% Sys: 0.000 ms/i 9%  
Entire network performance: Real: 87.011 ms/i User: 87.005 ms/i Sys: 0.001 ms/i  
Accounted by ops: Real: 83.129 ms/i 96% User: 83.127 ms/i 96% Sys: 0.001 ms/i 100%

Most time consuming operations:  
**Conv::DepthWise<F32>** 17 calls: Real: 61.049 ms/i 73% User: 61.048 ms/i 73% Sys: 0.000 ms/i 0%  
**MatMulUnaryImplASimpleB** 35 calls: Real: 11.925 ms/i 14% User: 11.925 ms/i 14% Sys: 0.000 ms/i 0%  
**Add::UnaryA** 36 calls: Real: 3.455 ms/i 4% User: 3.453 ms/i 4% Sys: 0.001 ms/i 87%  
**tf.FusedBatchNorm** 17 calls: Real: 3.286 ms/i 4% User: 3.285 ms/i 4% Sys: 0.000 ms/i 4%  
**Relu6** 35 calls: Real: 2.592 ms/i 3% User: 2.592 ms/i 3% Sys: 0.000 ms/i 9%  
Entire network performance: Real: 87.011 ms/i User: 87.005 ms/i Sys: 0.001 ms/i  
Accounted by ops: Real: 83.129 ms/i 96% User: 83.127 ms/i 96% Sys: 0.001 ms/i 100%

+-----+		+-----+	
220	Operation: <b>MatMulUnaryImplASimpleB</b>	Name: MobilenetV2/Logits/Conv2d_1c_1x1/Conv2D	
+-----+		+-----+	
	Input 0: Node #219/0 1x1x1x1792xF32		
+-----+		+-----+	
	Output 0: 1x1x1x1001xF32		
+-----+		+-----+	
	MM m:1 k:1792 n:1001 fma(16x6)		
+-----+		+-----+	
	Attr dilations: list {i: 1 i: 1 i: 1 i: 1}		
	Attr use_cudnn_on_gpu: b: true		
	Attr T: type: DT_FLOAT		
	Attr data_format: s: "NHWC"		
	Attr padding: s: "SAME"		
	Attr strides: list {i: 1 i: 1 i: 1 i: 1}		
+-----+		+-----+	
	FMA(F32) 1793792		
+-----+		+-----+	

# A few critical optimisation



**Machine independent  
classical matrix product optims**  
- packing  
- unrolling  
- im2col optim for convolution

**VFP2 mat mul kernel  
(unsafe)**

**NEON mat mul kernel  
(unsafe)**

**Rational expression  
for Sigmoid and Tanh**

**Experimental direct convolution  
(unsafe)**

**Cortex A53 (Raspberry Pi 3 B 1.2, Raspbian)**

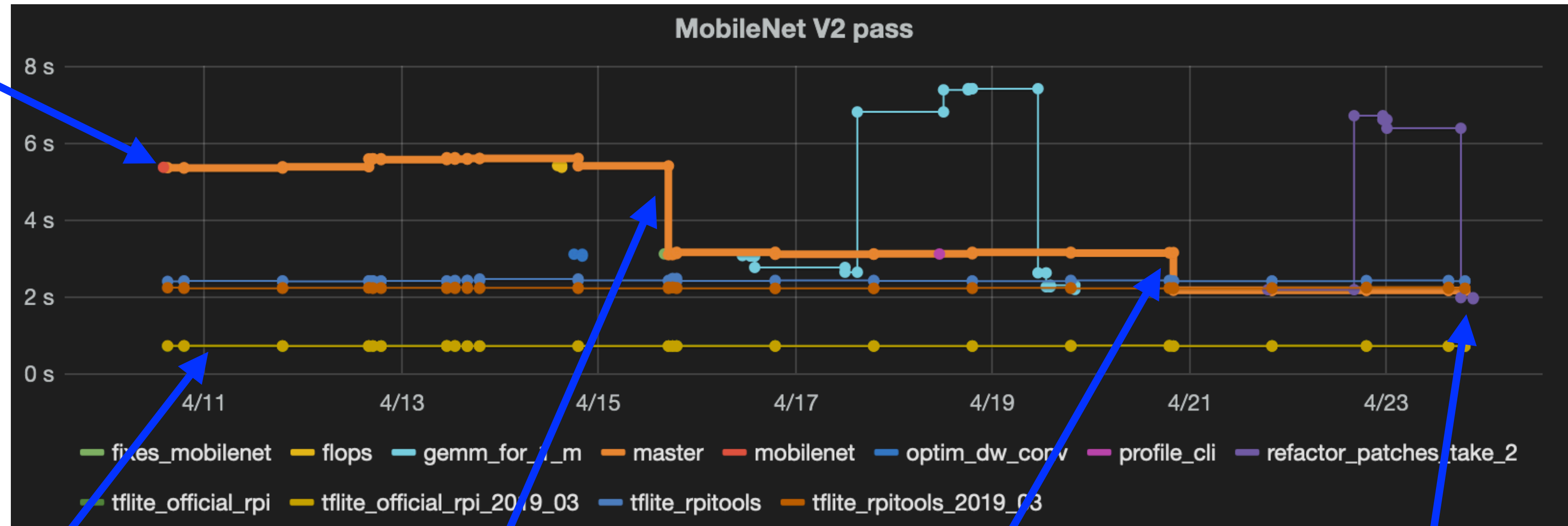


# What's next



# WIP: MobileNet v2 and DepthWise Conv

Naive  
depthwise  
conv impl

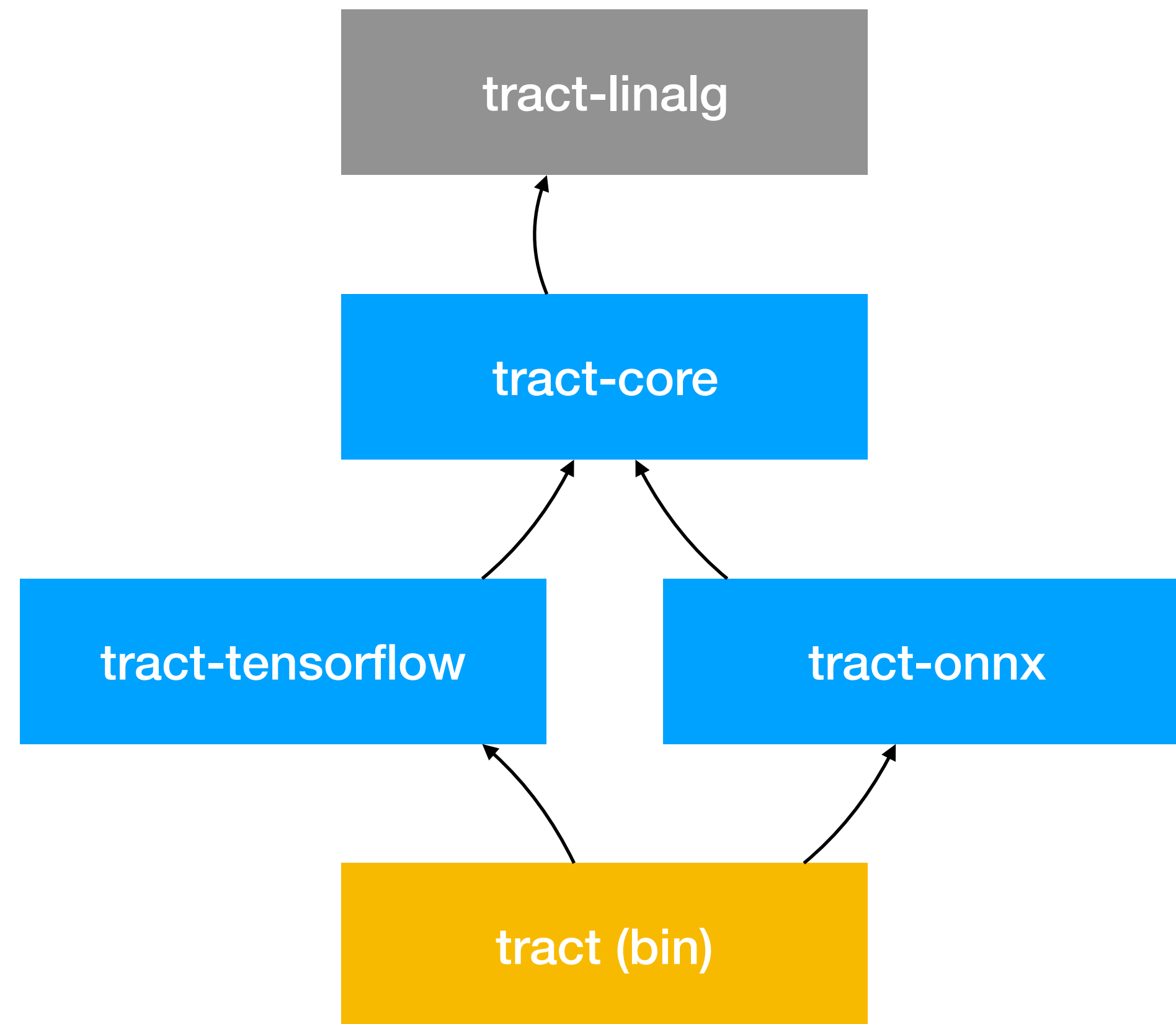


TensorFlow Lite

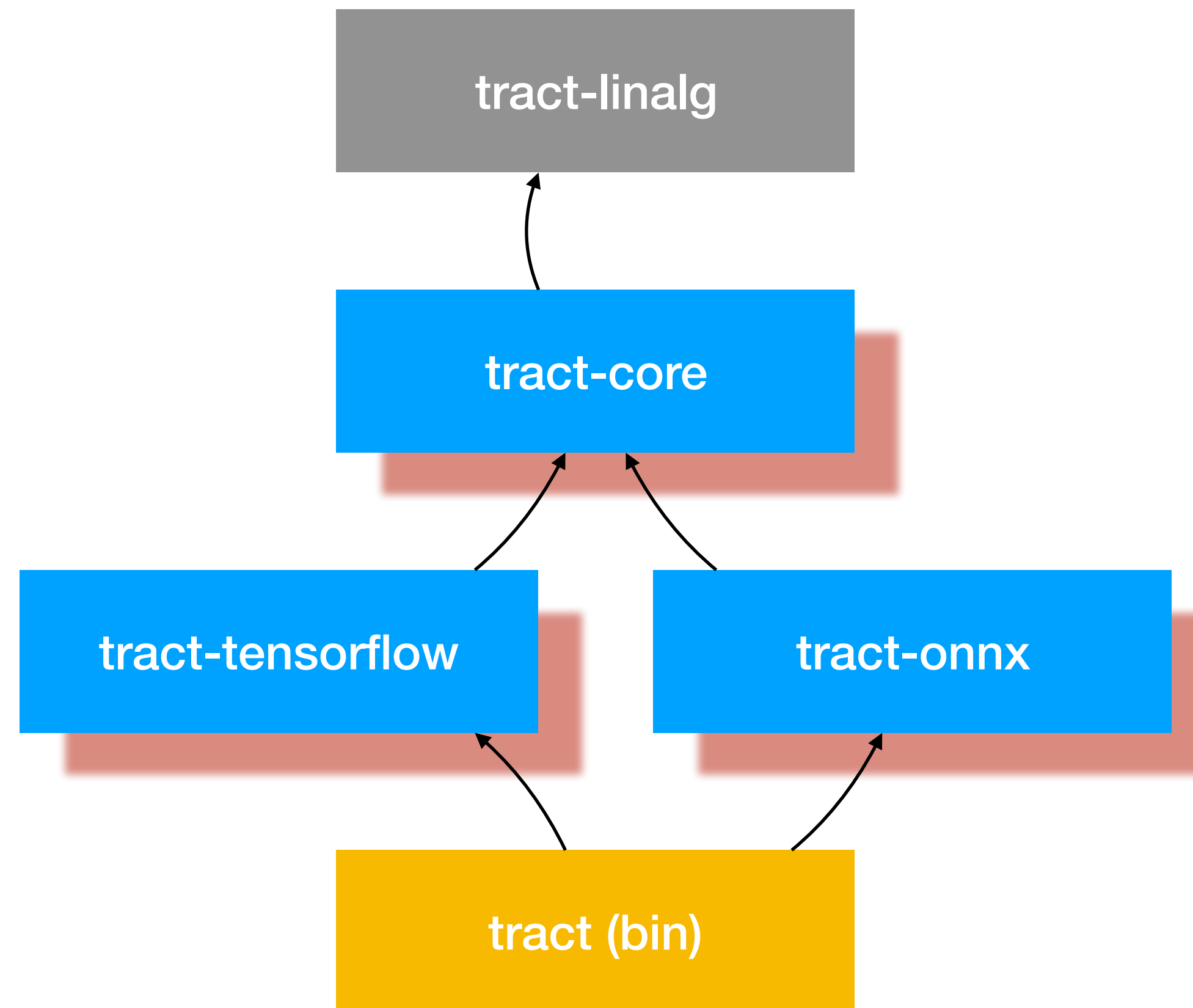
operator high-level  
optimisation  
(mostly safe)

introduce vector-matrix  
generic kernel  
(a tiny bit unsafe)

refactoring data access  
(very unsafe code)

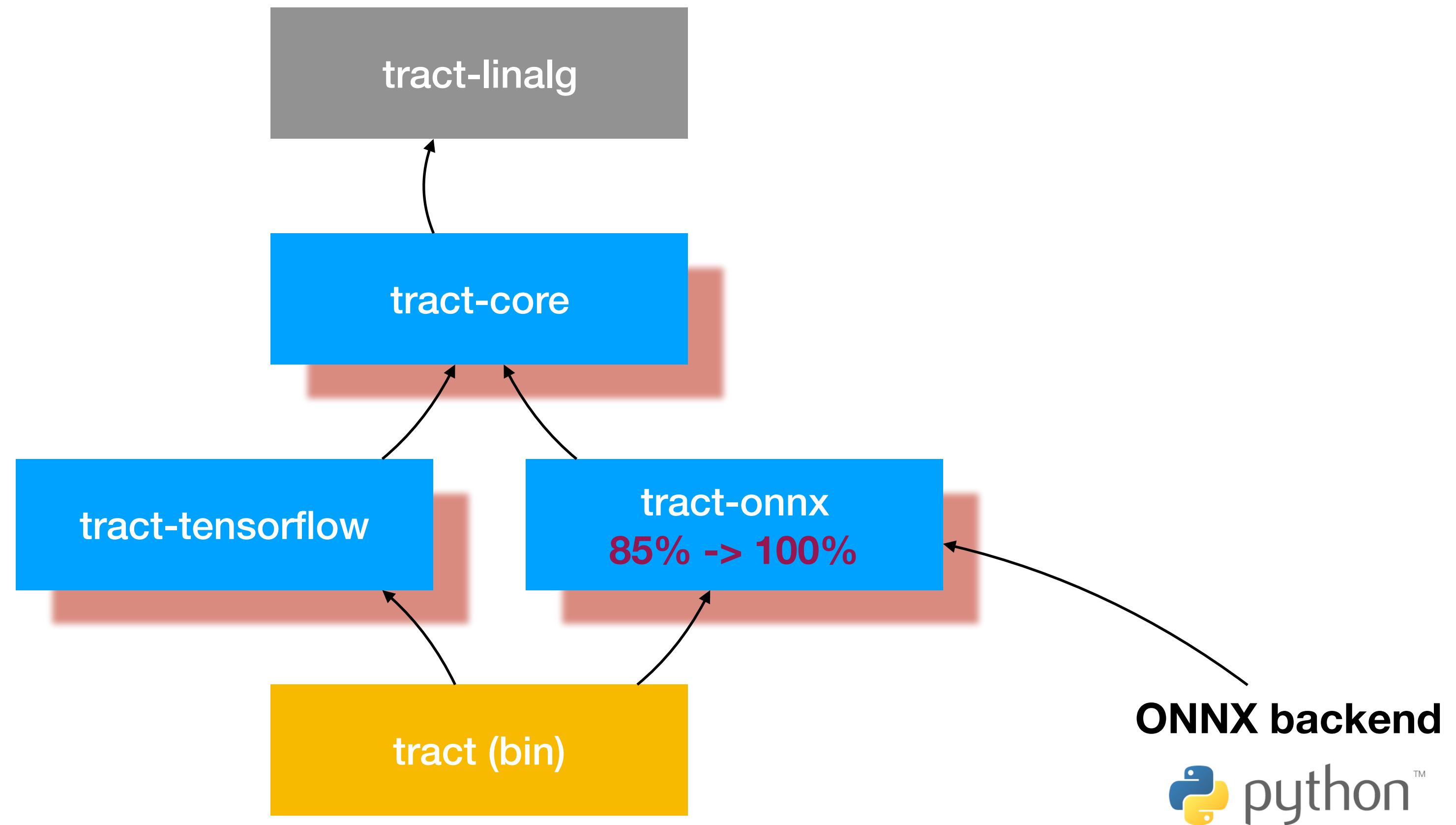


# FFI interfaces

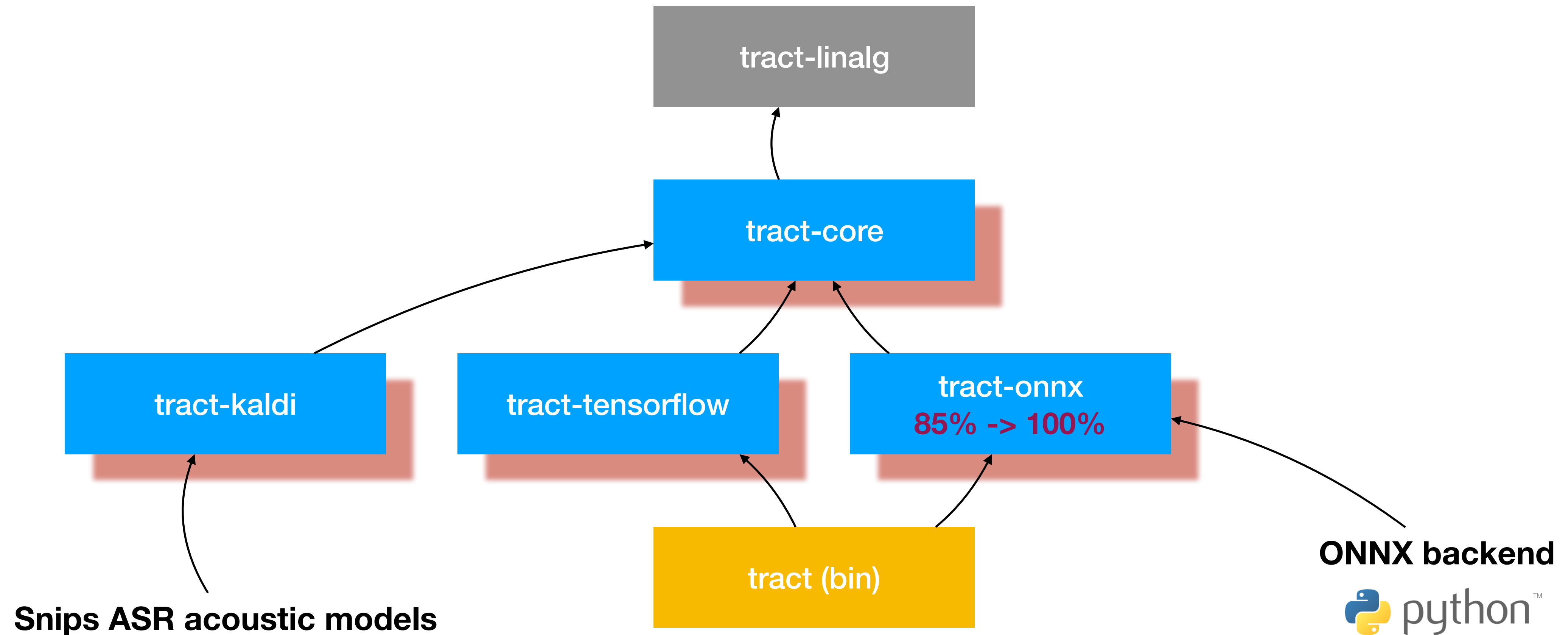




# ONNX full coverage and backend interface



# Kaldi support



## But also...

- Be as good as TensorFlow-Lite on as many devices/networks combinations as possible
- unsafe code is good !
- Reason more as an interpreter:
  - $\text{ReLu}(x) = \text{Max}(x, 0)$ ,  $\text{ReLu6}(x) = \text{Min}(\text{Max}(x, 0), 6)$ ,  $\text{ReLu20}$
  - sigmoid, tanh, erf:  
 $\text{polynomial}(x, \text{coefs1}) / \text{polynomial}(x, \text{coefs2})$