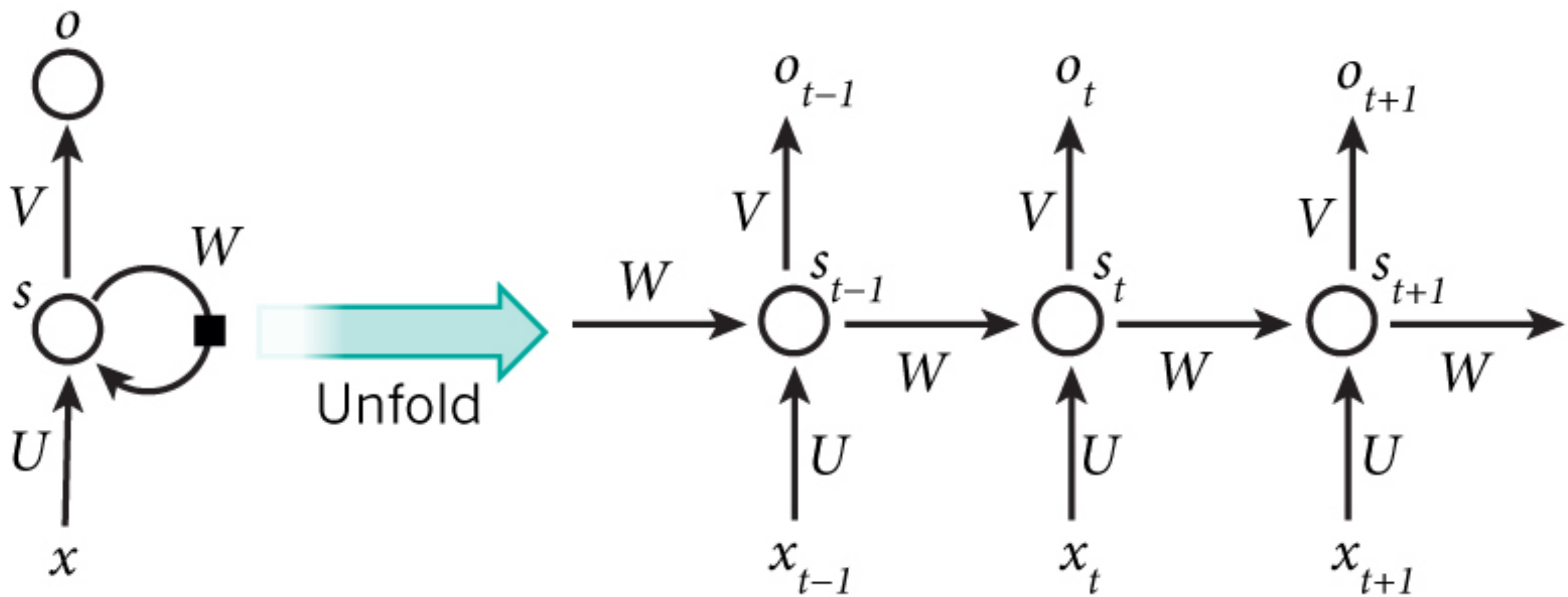
Neuronale Netze
Introspection

Prof. Dr.-Ing. Sebastian Stober

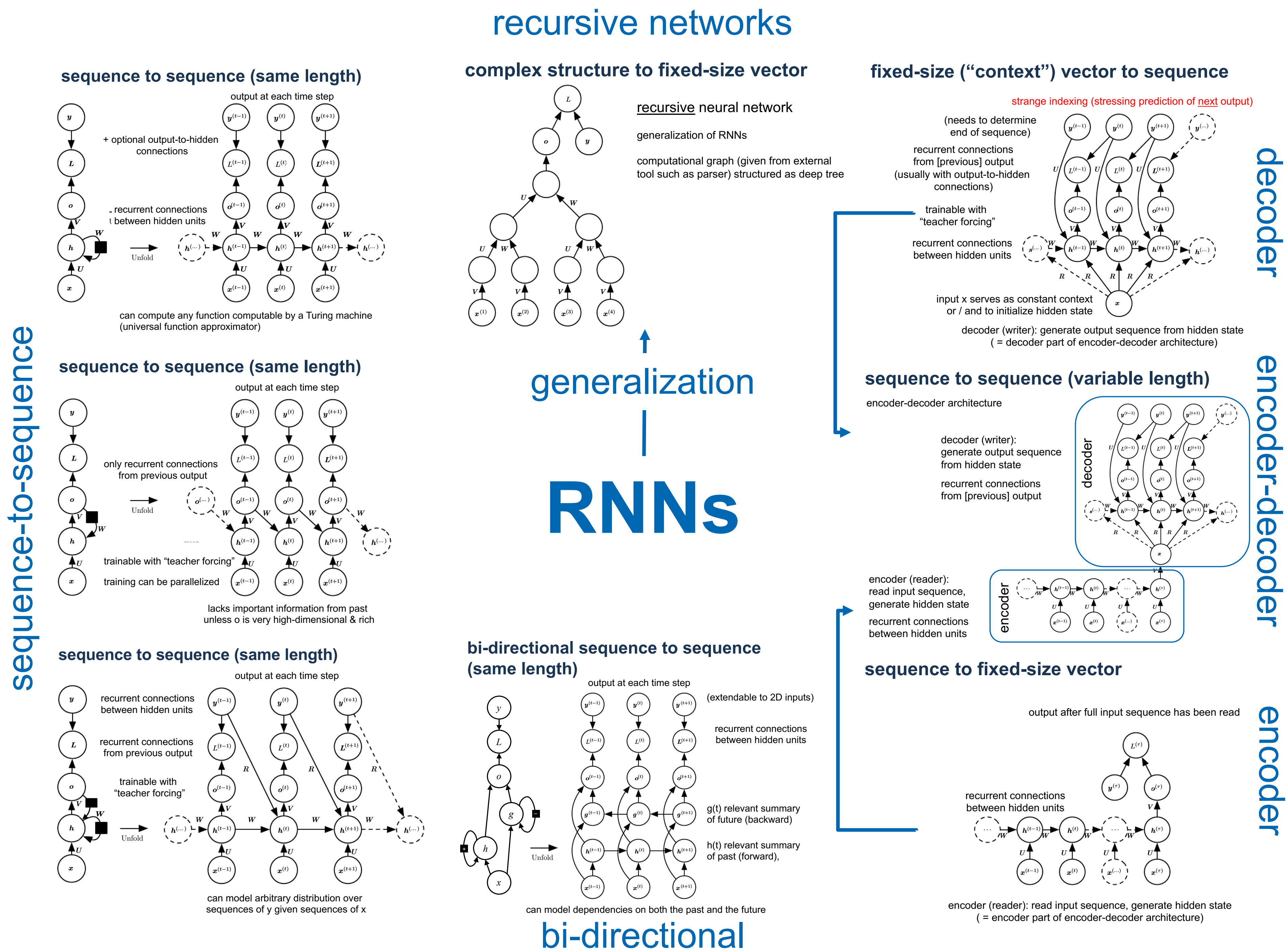
Artificial Intelligence Lab
Institut für Intelligente Kooperierende Systeme
Fakultät für Informatik
stober@ovgu.de

Recap

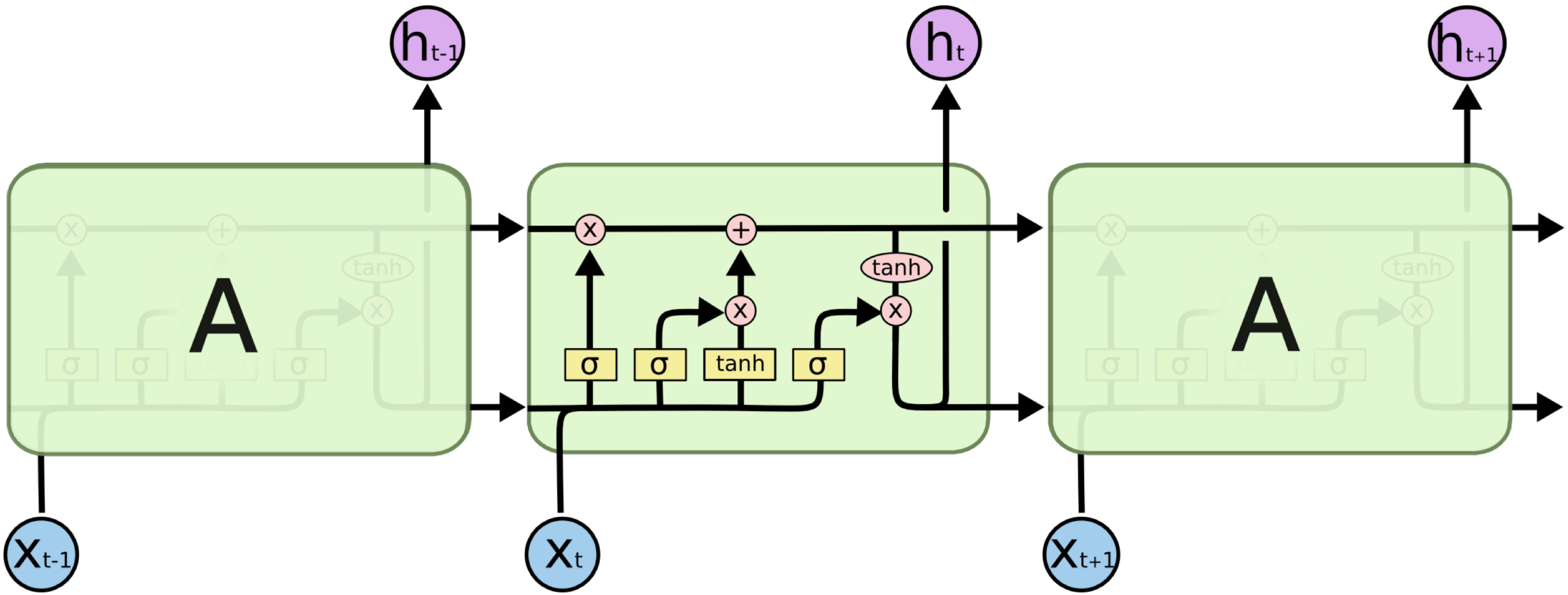
Recurrent Neural Nets



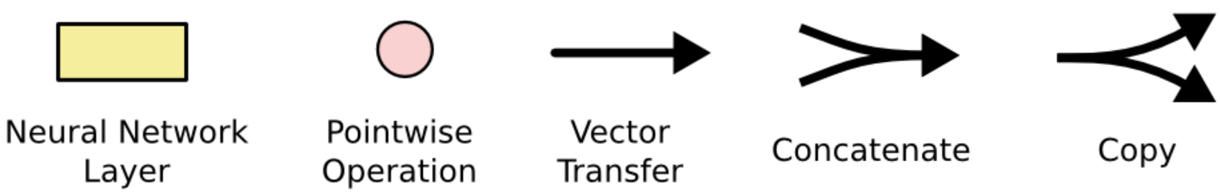
LeCun, Bengio & Hinton. "Deep Learning." *nature* 521.7553 (2015)


recursive networks
complex structure to fixed-size vector
recursive neural network
generalization of RNNs
computational graph (given from external tool such as parser) structured as deep tree
generalization
RNNs
sequence-to-sequence
sequence to sequence (same length)
output at each time step
+ optional output-to-hidden connections
recurrent connections between hidden units
Unfold
can compute any function computable by a Turing machine (universal function approximator)
sequence to sequence (same length)
output at each time step
only recurrent connections from previous output
Unfold
trainable with "teacher forcing"
training can be parallelized
lacks important information from past unless o is very high-dimensional & rich
sequence to sequence (same length)
output at each time step
recurrent connections between hidden units
recurrent connections from previous output
trainable with "teacher forcing"
Unfold
can model arbitrary distribution over sequences of y given sequences of x
bi-directional sequence to sequence (same length)
output at each time step
(extendable to 2D inputs)
recurrent connections between hidden units
g(t) relevant summary of future (backward)
h(t) relevant summary of past (forward),
Unfold
can model dependencies on both the past and the future
bi-directional
fixed-size ("context") vector to sequence
strange indexing (stressing prediction of next output)
(needs to determine end of sequence)
recurrent connections from [previous] output (usually with output-to-hidden connections)
trainable with "teacher forcing"
recurrent connections between hidden units
input x serves as constant context or / and to initialize hidden state
decoder (writer): generate output sequence from hidden state (= decoder part of encoder-decoder architecture)
decoder
sequence to sequence (variable length)
encoder-decoder architecture
decoder (writer): generate output sequence from hidden state
recurrent connections from [previous] output
decoder
encoder (reader): read input sequence, generate hidden state
recurrent connections between hidden units
encoder
encoder-decoder
sequence to fixed-size vector
output after full input sequence has been read
recurrent connections between hidden units
encoder (reader): read input sequence, generate hidden state (= encoder part of encoder-decoder architecture)
encoder

LSTM



The repeating module in an LSTM contains four interacting layers.



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Generate Image Captions

Describes without errors

A person riding a motorcycle on a dirt road.

A group of young people playing a game of frisbee.

A herd of elephants walking across a dry grass field.

Describes with minor errors

Two dogs play in the grass.

Two hockey players are fighting over the puck.

A close up of a cat laying on a couch.

Somewhat related to the image

A skateboarder does a trick on a ramp.

A little girl in a pink hat is blowing bubbles.

A red motorcycle parked on the side of the road.

Unrelated to the image

A dog is jumping to catch a frisbee.

A refrigerator filled with lots of food and drinks.

A yellow school bus parked in a parking lot.

Answer Visual Questions

What vegetable is on the plate?
Neural Net: broccoli
Ground Truth: broccoli

What color are the shoes on the person's feet ?
Neural Net: brown
Ground Truth: brown

How many school busses are there?
Neural Net: 2
Ground Truth: 2

What sport is this?
Neural Net: baseball
Ground Truth: baseball

What is on top of the refrigerator?
Neural Net: magnets
Ground Truth: cereal

What uniform is she wearing?
Neural Net: shorts
Ground Truth: girl scout

What is the table number?
Neural Net: 4
Ground Truth:40

What are people sitting under in the back?
Neural Net: bench
Ground Truth: tent

https://avisingh599.github.io/deeplearning/visual-qa/

RNNs

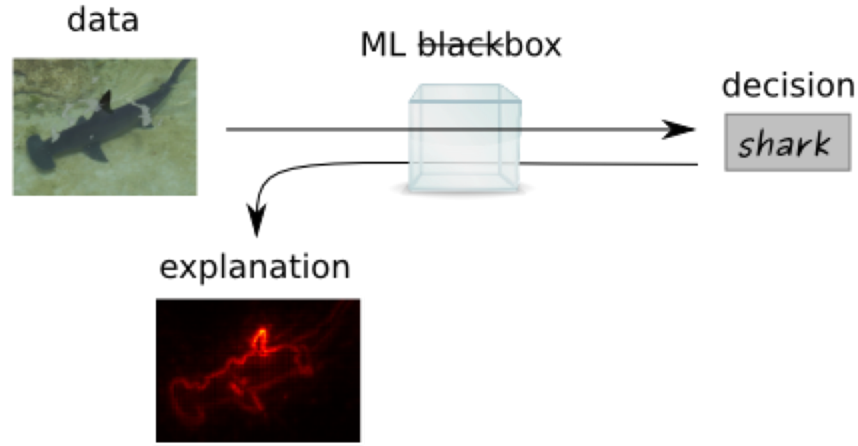
- work well for sequential data
  - time series (with low sampling rate)
  - texts (translation, discourse, sentiment, ...)
- support variable-length input
  - including long-term dependencies
- are hard to parallelize

Introspection

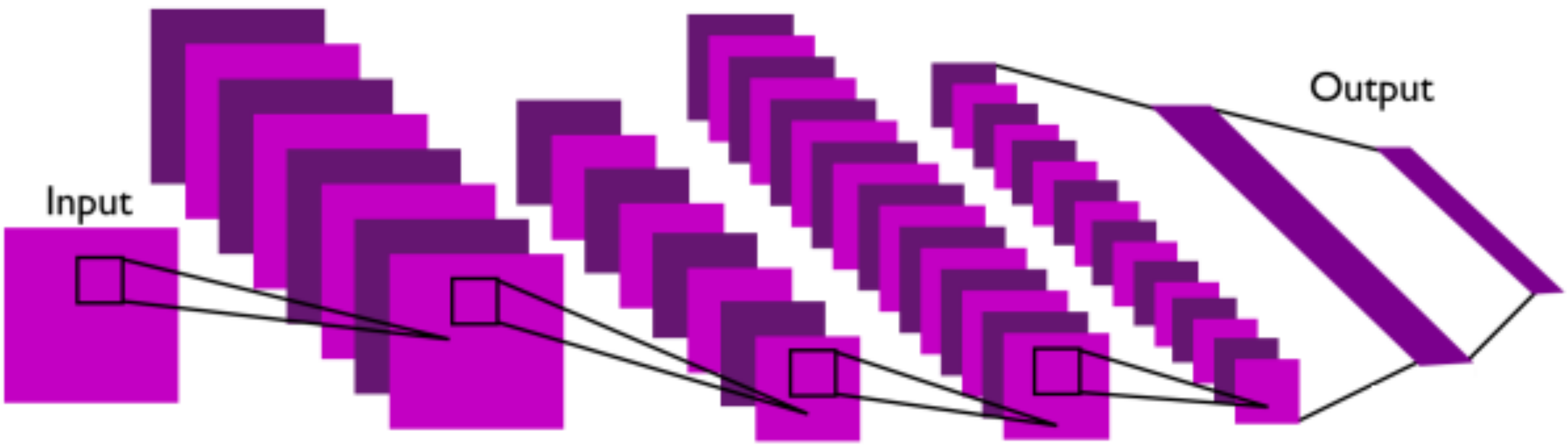
Types of Introspection

feature visualization

layer-wise relevance propagation (LRP)
deep Taylor decomposition



Feature Visualization

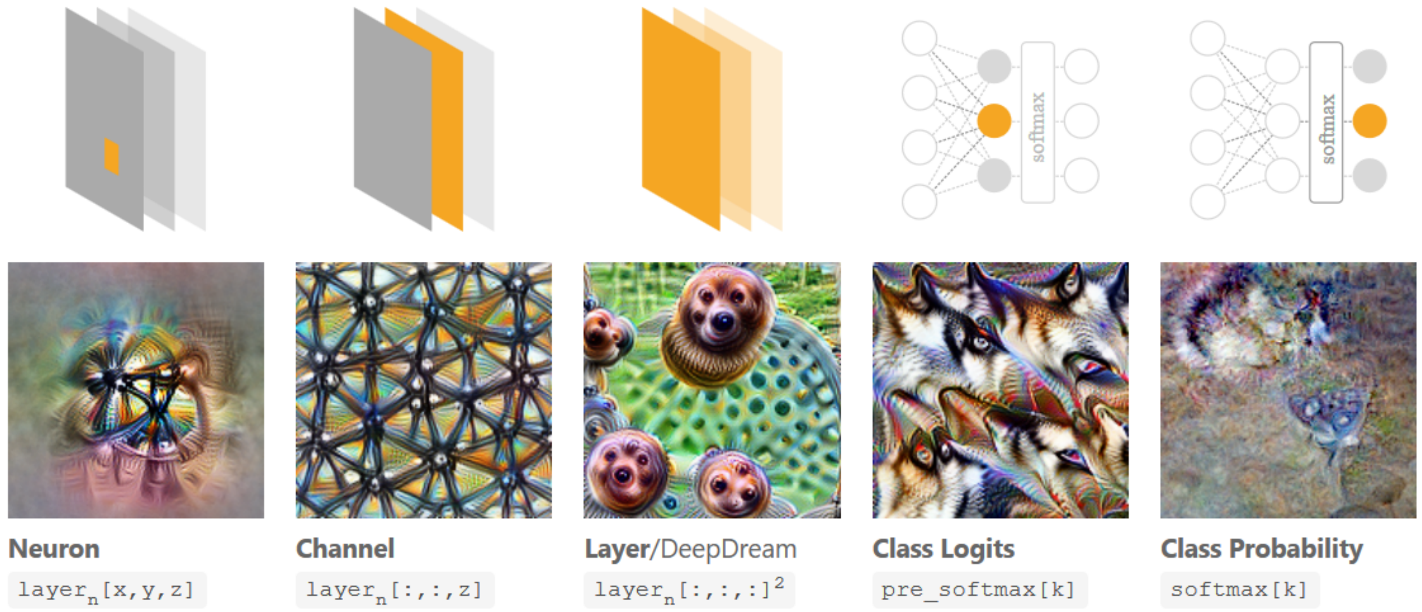


feature visualization by optimization
(find the input that optimizes a particular part of the network)

Feature Visualization

https://distill.pub/2017/feature-visualization/

Feature Visualization



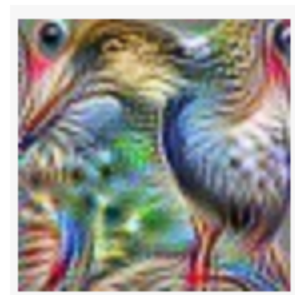
https://distill.pub/2017/feature-visualization/

Feature Visualization

What's the main problem with the (vanilla) optimization approach?
How do we solve this?

unregularized optimization is unnatural

vs



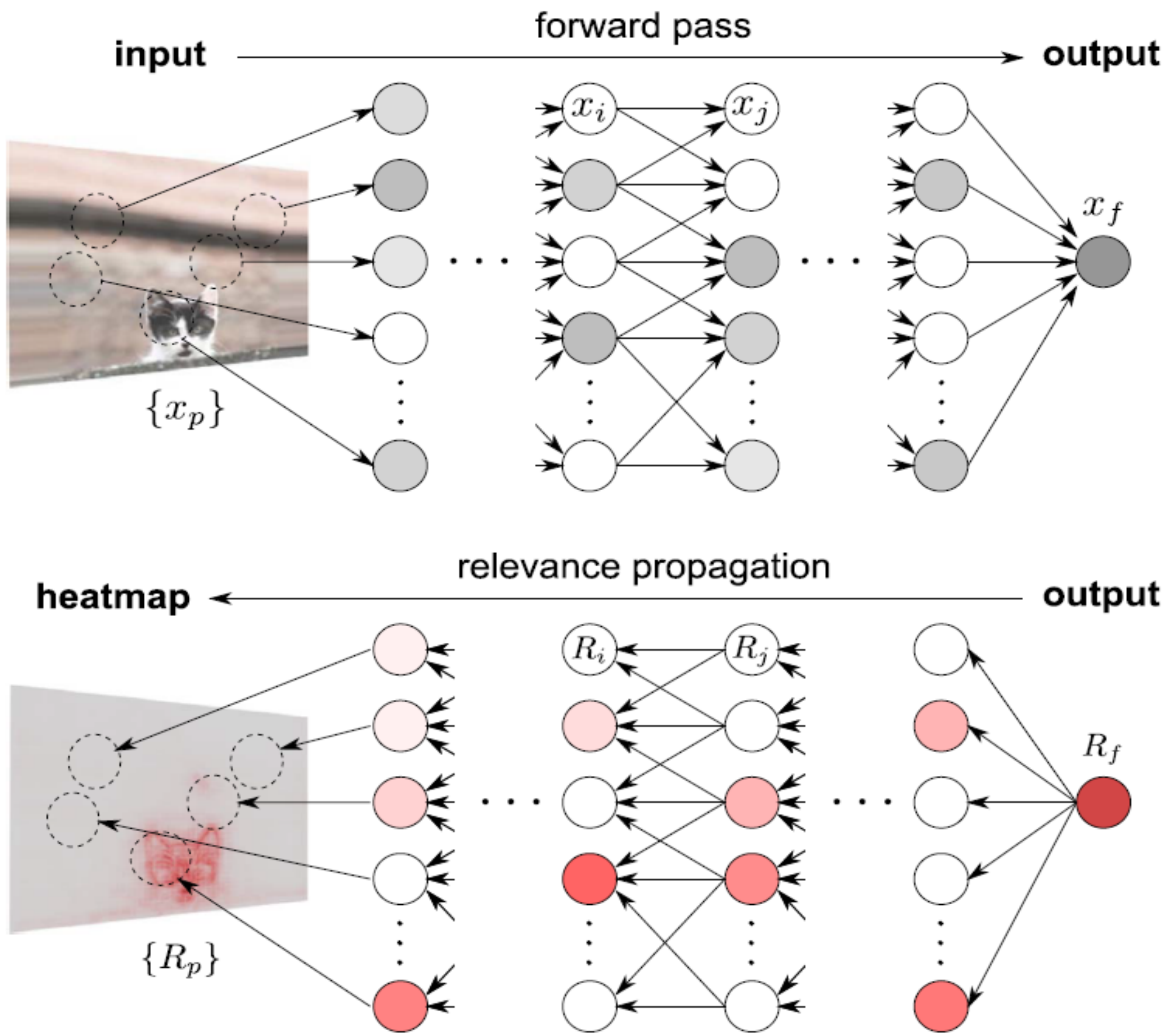
regularization methods

frequency penalization

transformation robustness

learned prior

Layer-wise Relevance Propagation (LRP)



[Montavon et al. (2017). Explaining nonlinear classification decisions with deep Taylor decomposition.]

Deep Taylor Decomposition and LRP

What's the difference?

deep Taylor decomposition

$$R_d^{(1)} = (x - x_0)_{(d)} \cdot \frac{\partial f}{\partial x_{(d)}}(x_0)$$

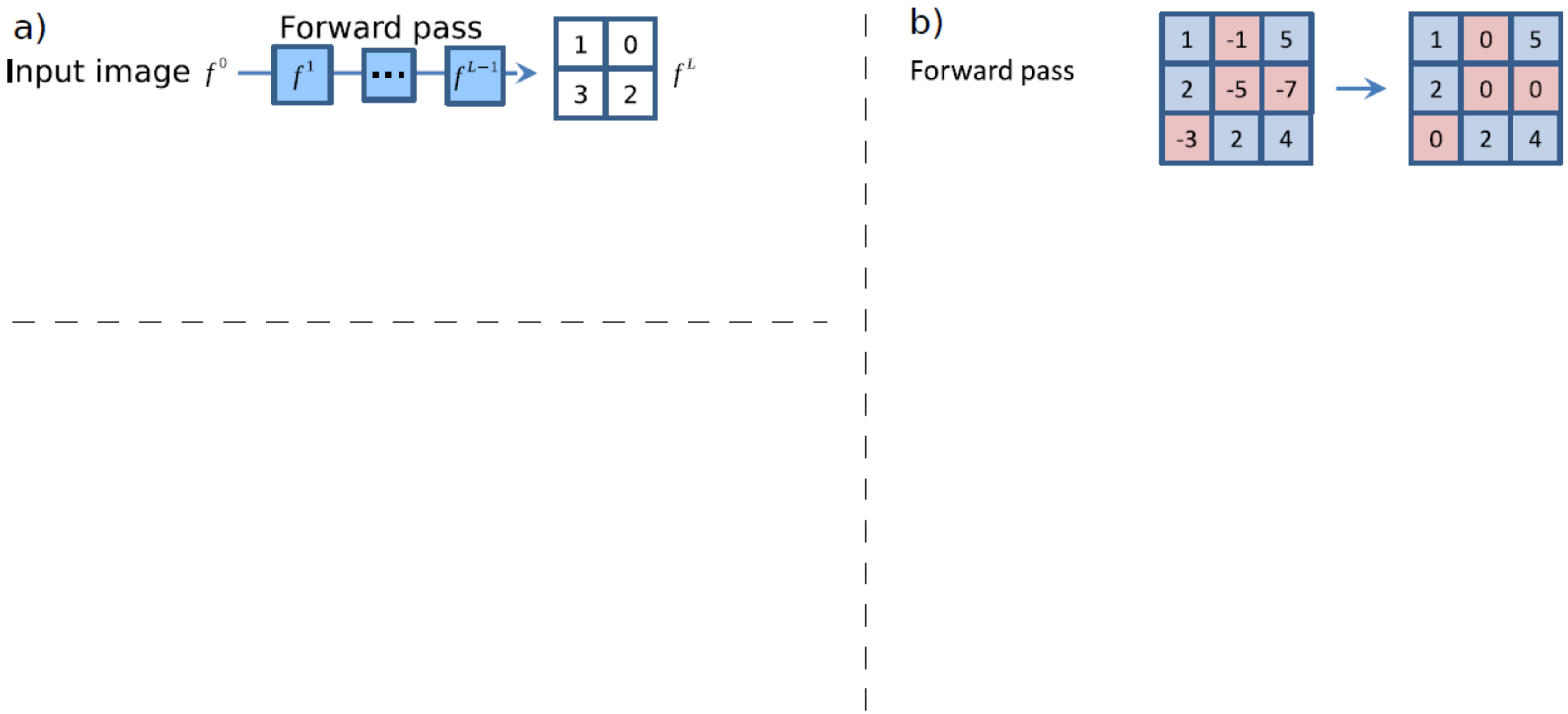
- root point x_0 must be determined
- computationally efficient (backprop)

layer-wise relevance propagation

$$R_{i \leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j} \cdot R_j^{(l+1)}$$

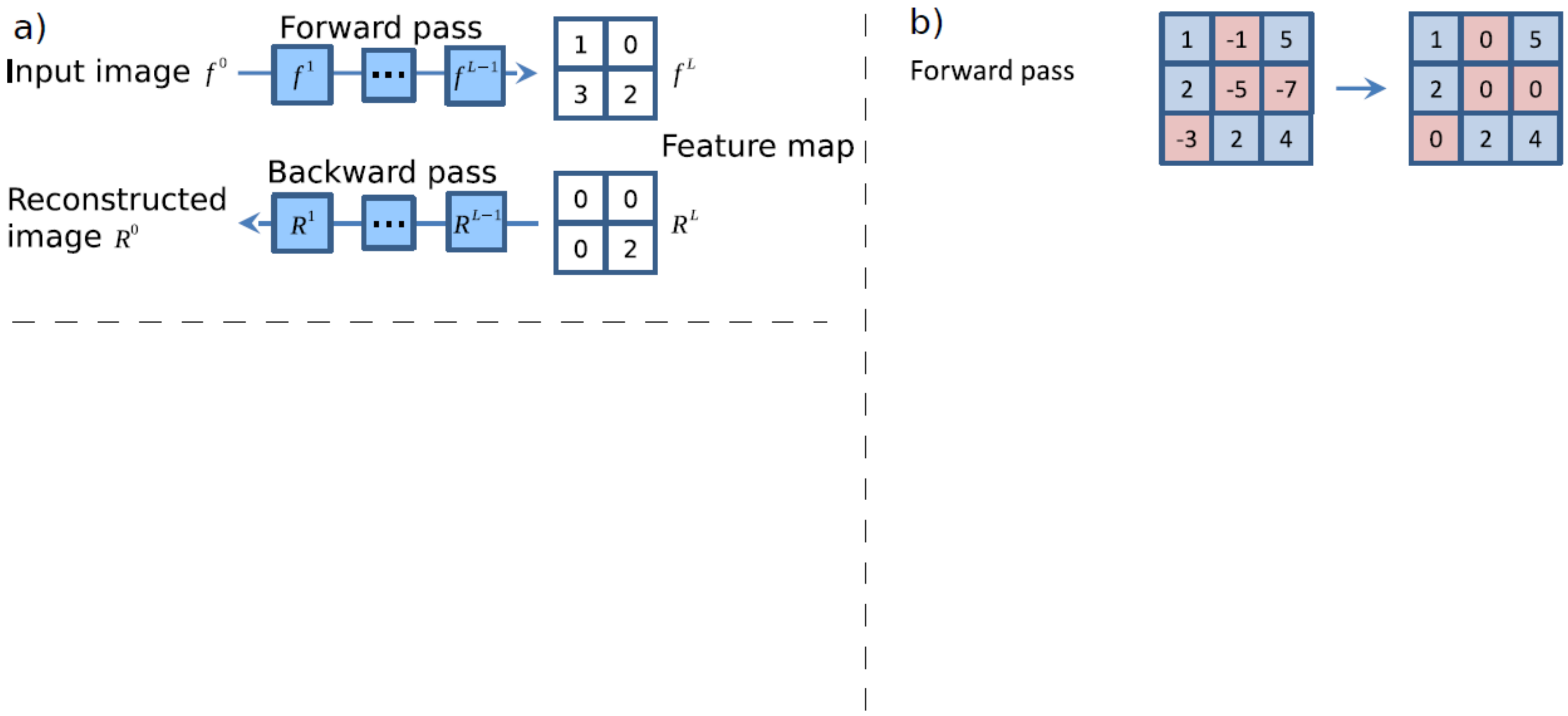
- no root point needed
- computationally expensive

Other Ways of Propagating Output Signals back to the Input



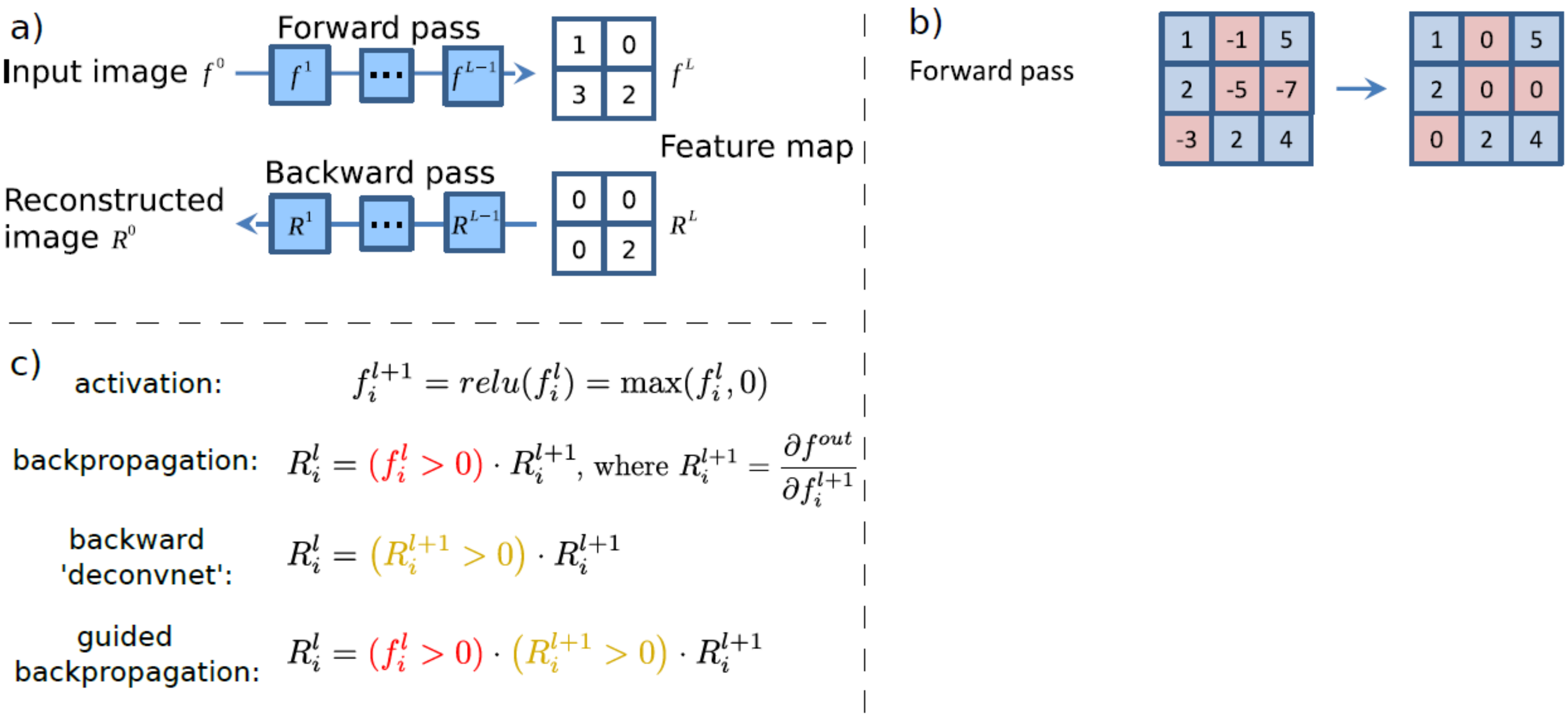
[Springenberg et al. (2014). Striving for Simplicity: The All Convolutional Net]

Other Ways of Propagating Output Signals back to the Input



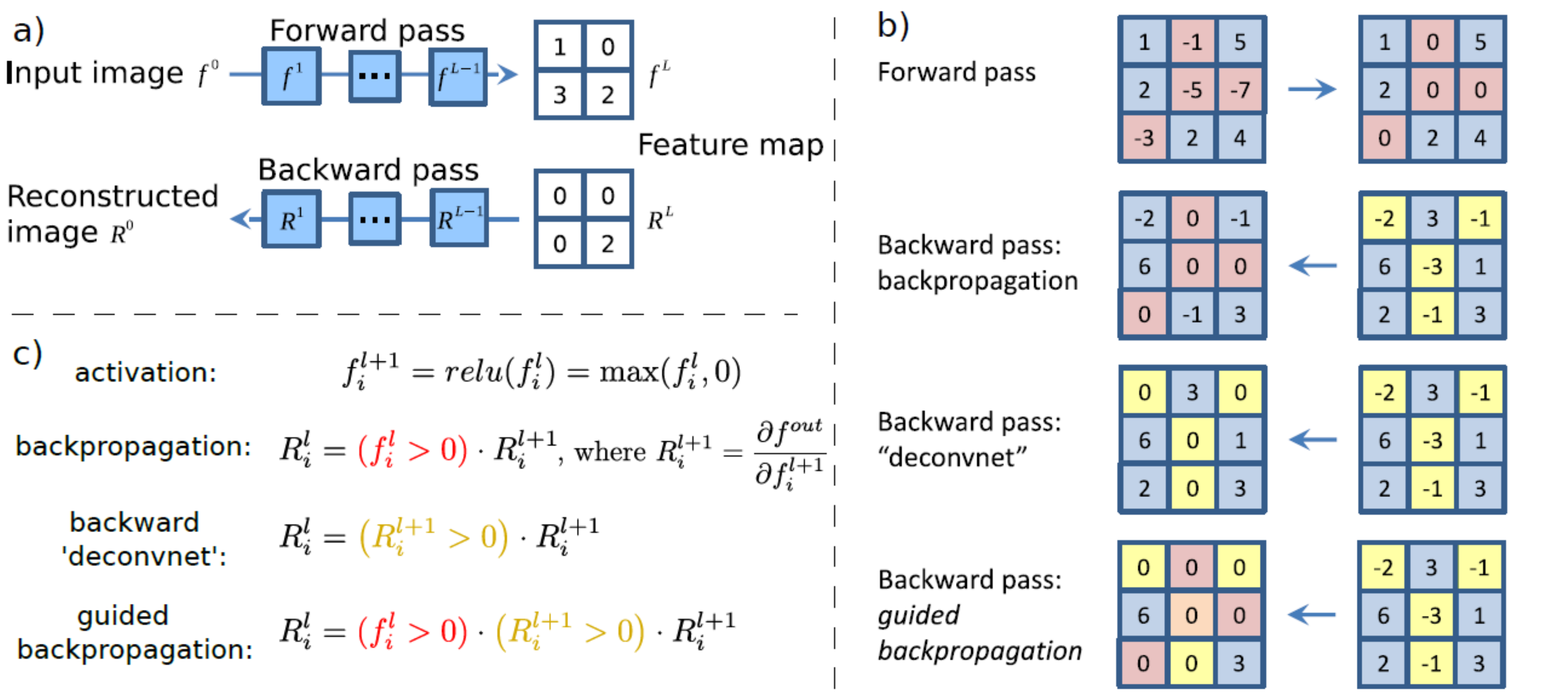
[Springenberg et al. (2014). Striving for Simplicity: The All Convolutional Net]

Other Ways of Propagating Output Signals back to the Input



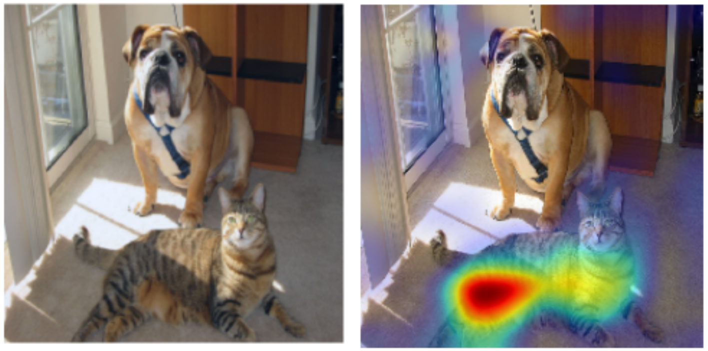
[Springenberg et al. (2014). Striving for Simplicity: The All Convolutional Net]

Other Ways of Propagating Output Signals back to the Input



[Springenberg et al. (2014). Striving for Simplicity: The All Convolutional Net]

GradCAM: Gradient-weighted Class Activation Mapping



(a) Original Image (c) Grad-CAM 'Cat'

$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i \sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

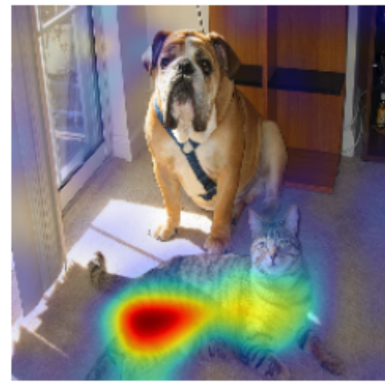
$$L^c_{\text{Grad-CAM}} = ReLU \underbrace{\left(\sum_k \alpha_k^c A^k\right)}_{\text{linear combination}}$$

importance of feature map A^k
for class c

combine all feature maps A^k
in one layer as weighted sum

GradCAM: Gradient-weighted Class Activation Mapping

(a) Original Image



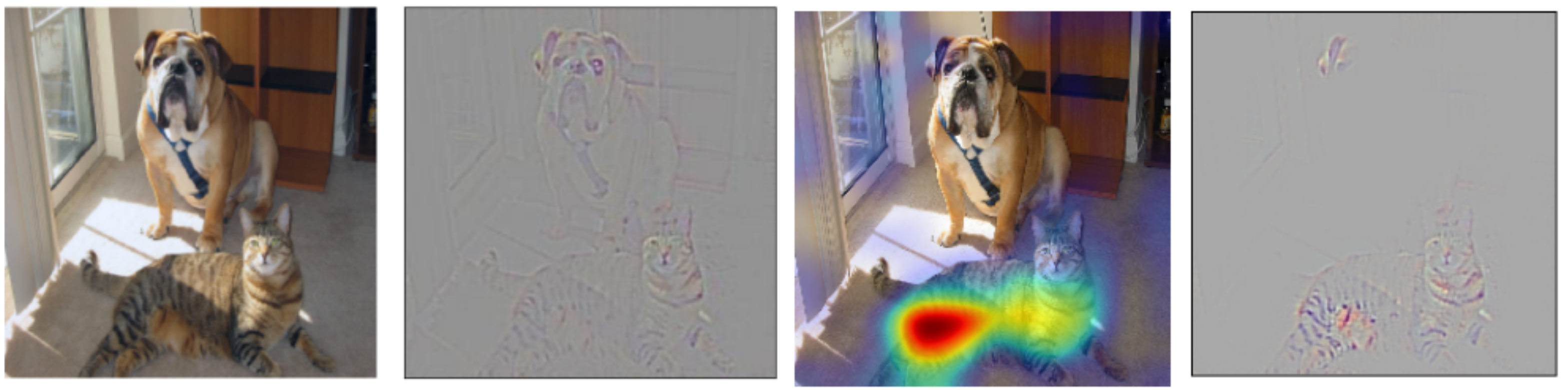
(c) Grad-CAM 'Cat'

(g) Original Image

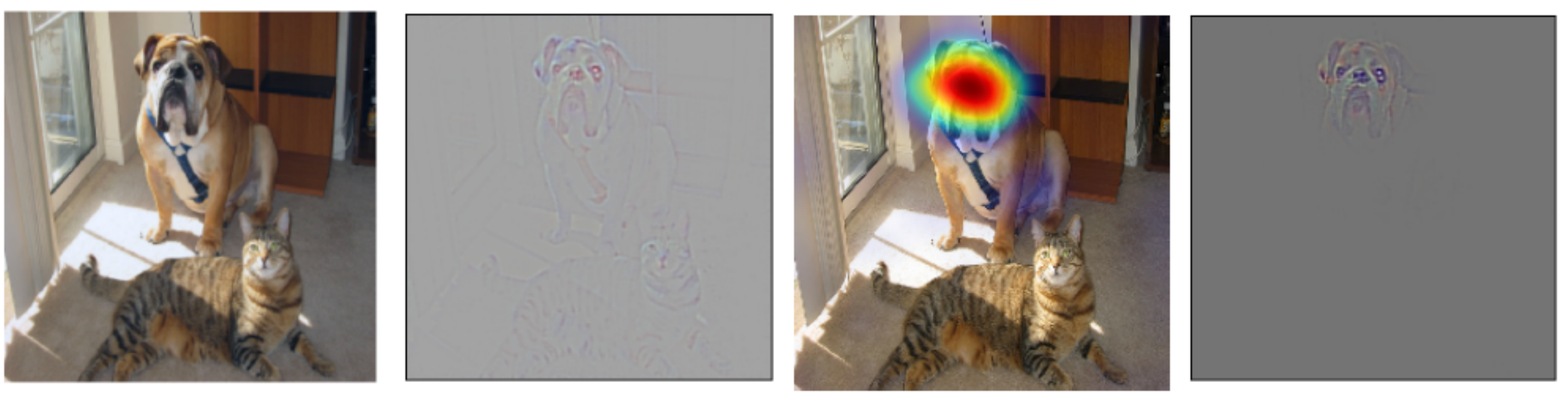
(i) Grad-CAM 'Dog'

[Selvaraju et al. (2016). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization.]

GradCAM: Gradient-weighted Class Activation Mapping



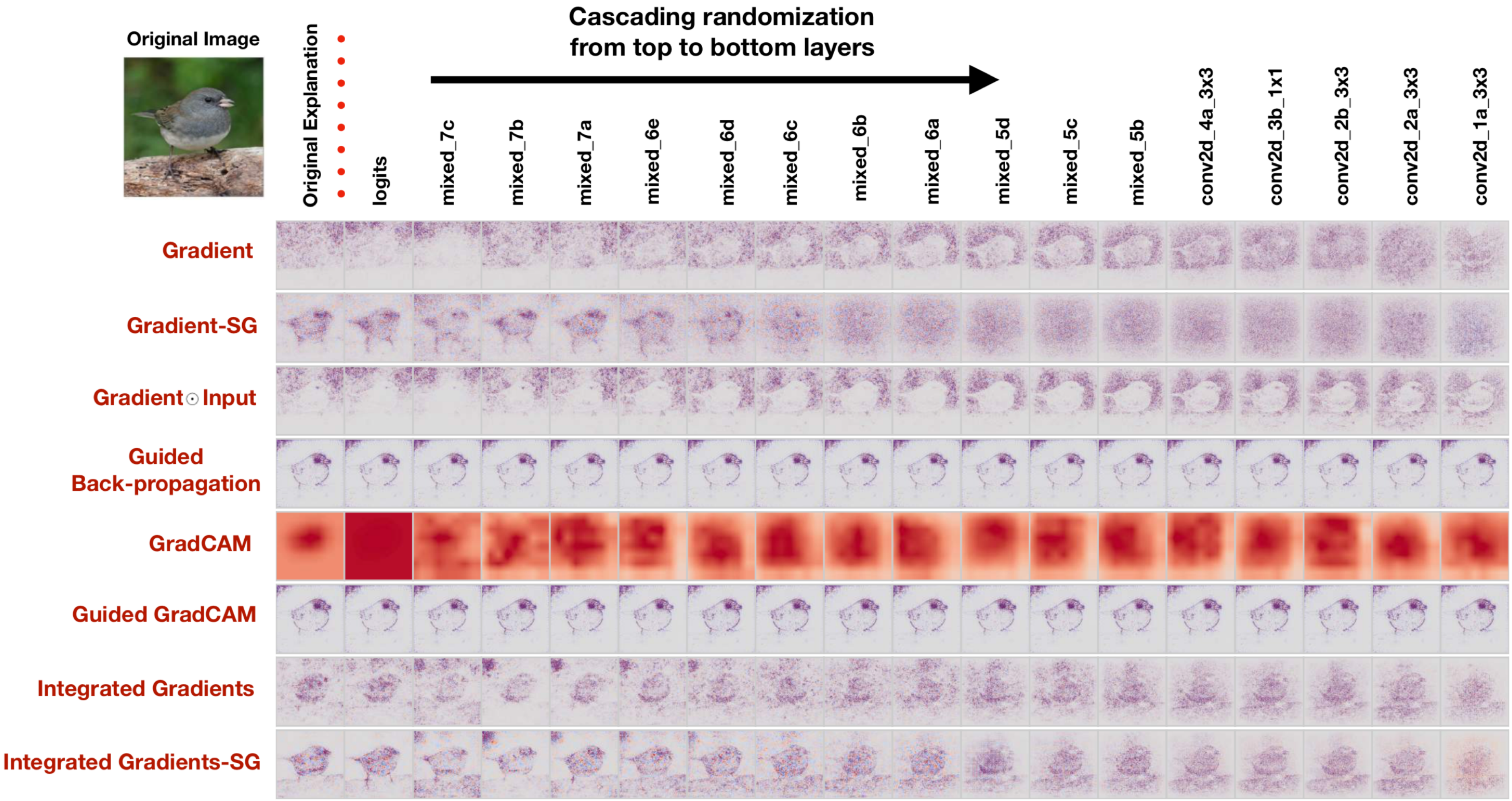
(a) Original Image (b) Guided Backprop ‘Cat’ (c) Grad-CAM ‘Cat’ (d) Guided Grad-CAM ‘Cat’



(g) Original Image (h) Guided Backprop ‘Dog’ (i) Grad-CAM ‘Dog’ (j) Guided Grad-CAM ‘Dog’

[Selvaraju et al. (2016). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization.]

Sanity Checks for Introspection



[Adebayo et al. (2018). Sanity Checks for Saliency Maps.]

Problems

- these models
  - sometimes require particular architectures (e.g. only 2D-convolution with max-pooling)
  - mostly use ReLUs and a positive input space (which pixels positively influence an output class)
  - are mostly evaluated only for images (visually interpretable)
- not well applicable for
  - other activation functions (allowing negative activation)
  - real-valued input space (negative values)
  - visually hardly interpretable data (e.g. waveforms)

Introspection for Speech Processing Models

Speech Recognizer on a Budget

- data:
  - use only free / public datasets
- model with limited compute resources:
  - single (consumer-level) GPU for training
  - not more than a few days for training
  - real-time capability during deployment
- loss function

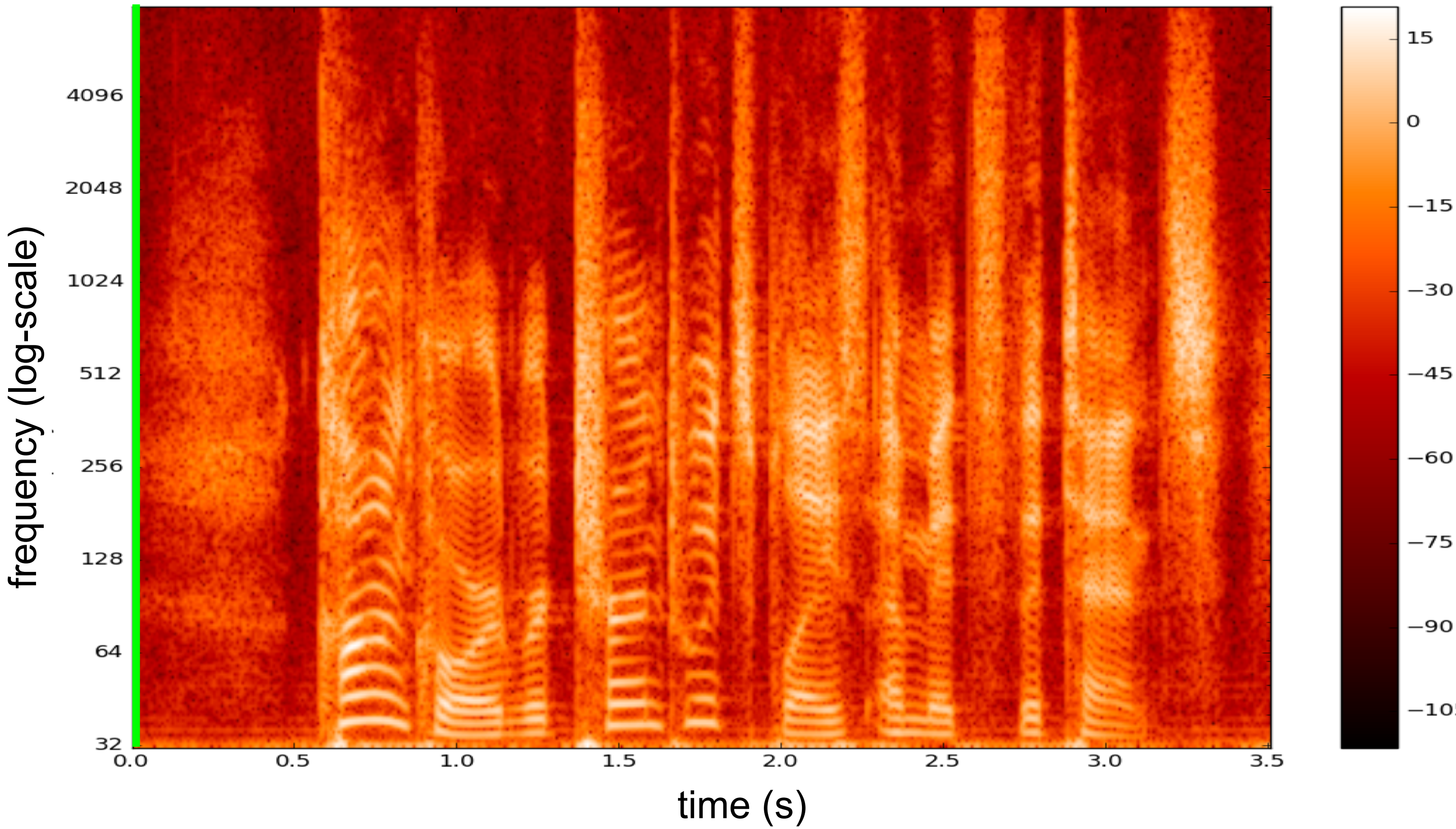
Training Data (English)

- LibriSpeech Corpus http://www.openslr.org/12/
  - ~1000h annotated audio
  - from public domain audio books
  - semi-automatically cut into phrases
  - good recording quality

"LibriSpeech: an ASR corpus based on public domain audio books", V. Panayotov, G. Chen, D. Povey and S. Khudanpur, ICASSP 2015

Input: Spectrogram

"Concord returned to its place amidst the tents."

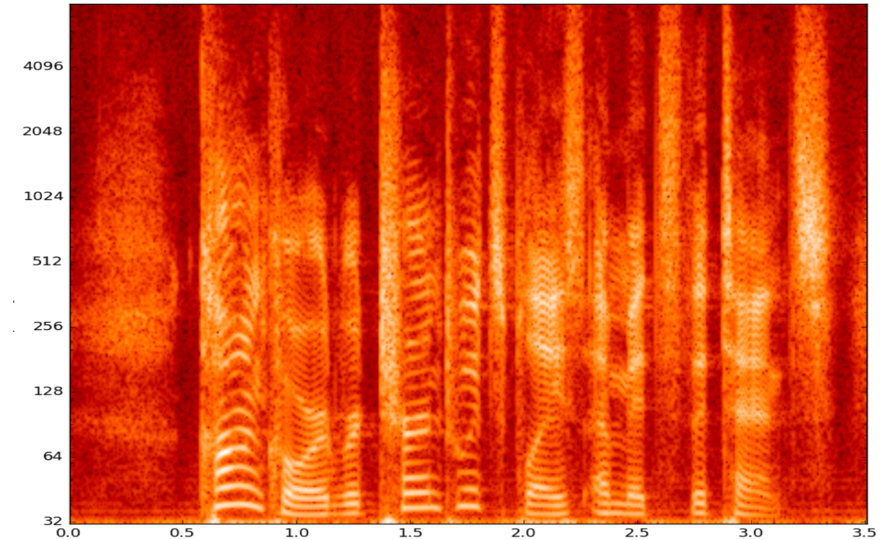


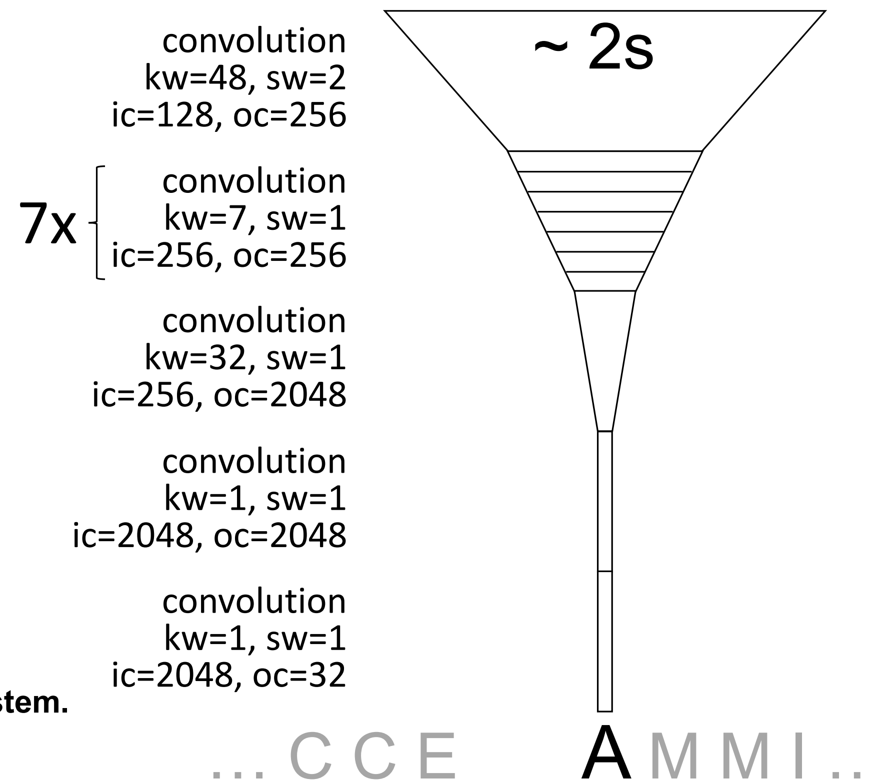
audio source: Alexandre Dumas "Ten Years Later", chapter 86 (LibriSpeech)

Wav2Letter
(Facebook AI, 2016)

- 11 CNN layers
- ~ 25 Mio parameters
- 50 letters / s

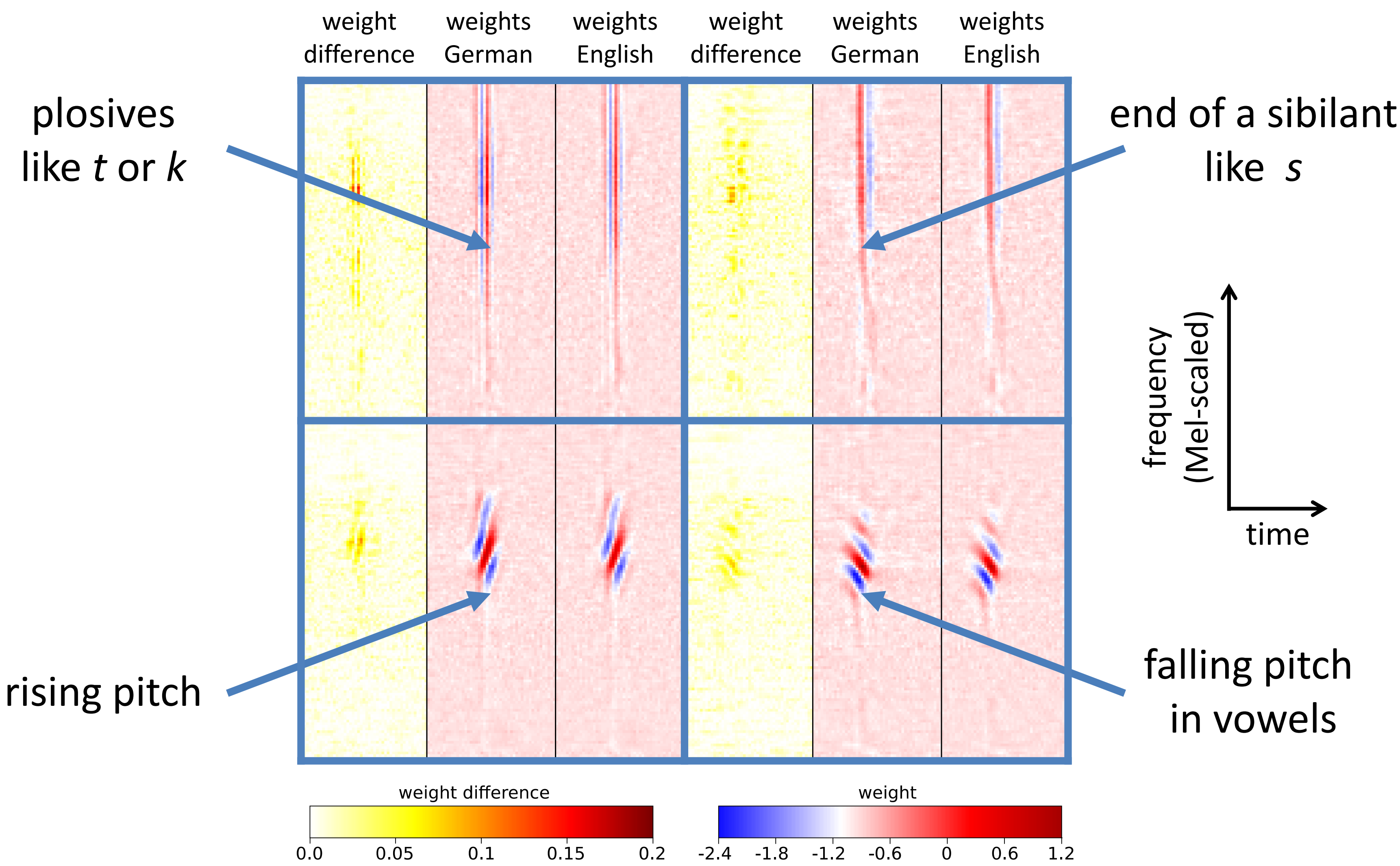
- 1-2 days of training (Geforce 1080 Ti)



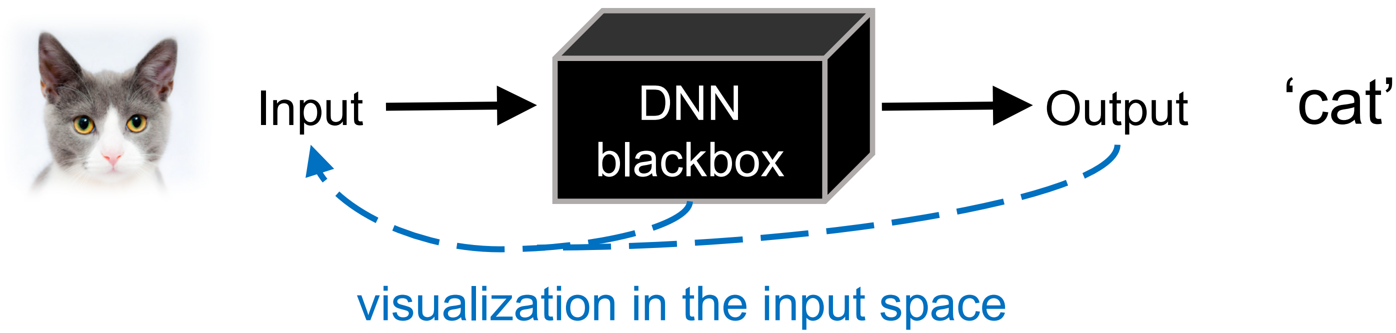


R. Collobert, C. Puhrsch & G. Synnaeve. 2016.
Wav2letter: an end-to-end convnet-based speech recognition system.
http://arxiv.org/abs/1609.03193

Learned Patterns (layer 1)



Typical Introspection Approaches

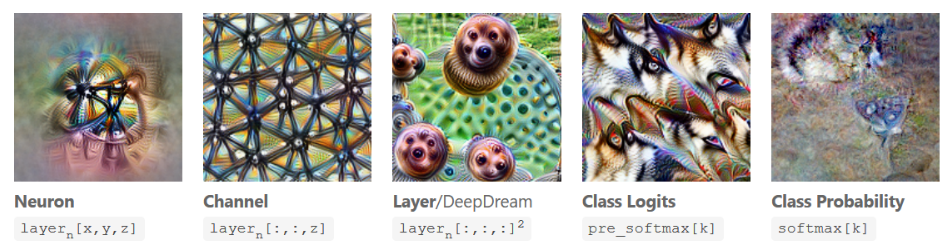


saliency maps
back-projecting the predicted class

(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat'

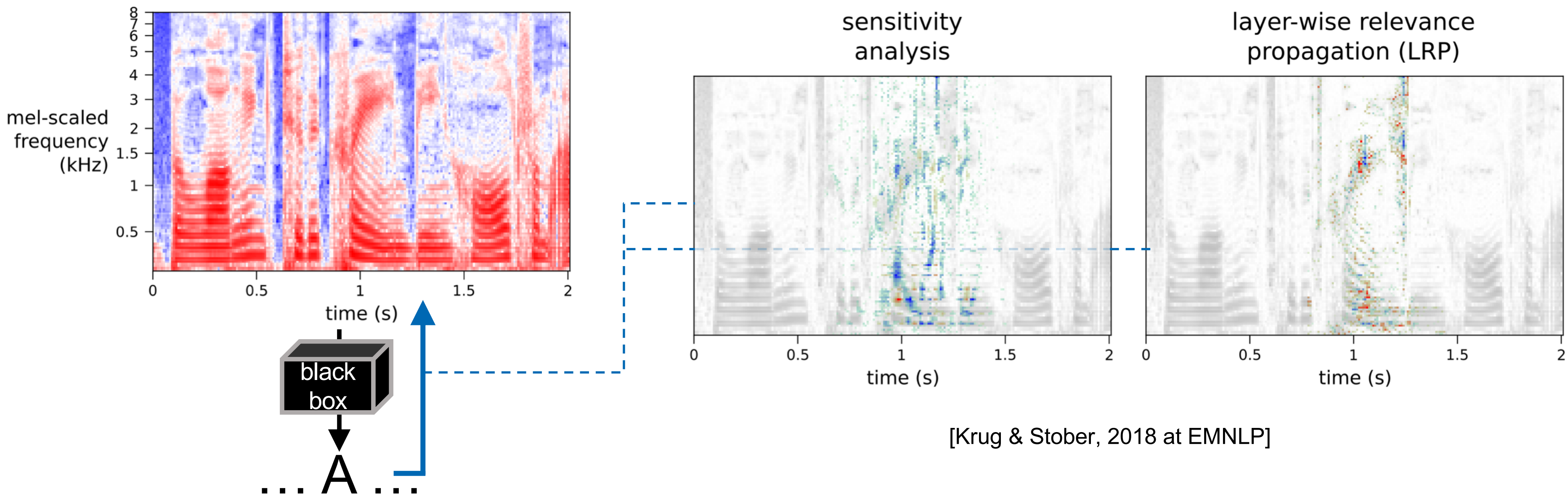
[Selvaraju et al., 2016]

activation maximization (AM)
Optimize input to maximally activate parts of network



[https://distill.pub/2017/feature-visualization/]

Does this also work for speech recognition?



[Krug & Stober, 2018 at EMNLP]

- **saliency maps on the input and activation maximization are not easily interpretable for speech**
- **audio is time series (of spectrogram frames)**

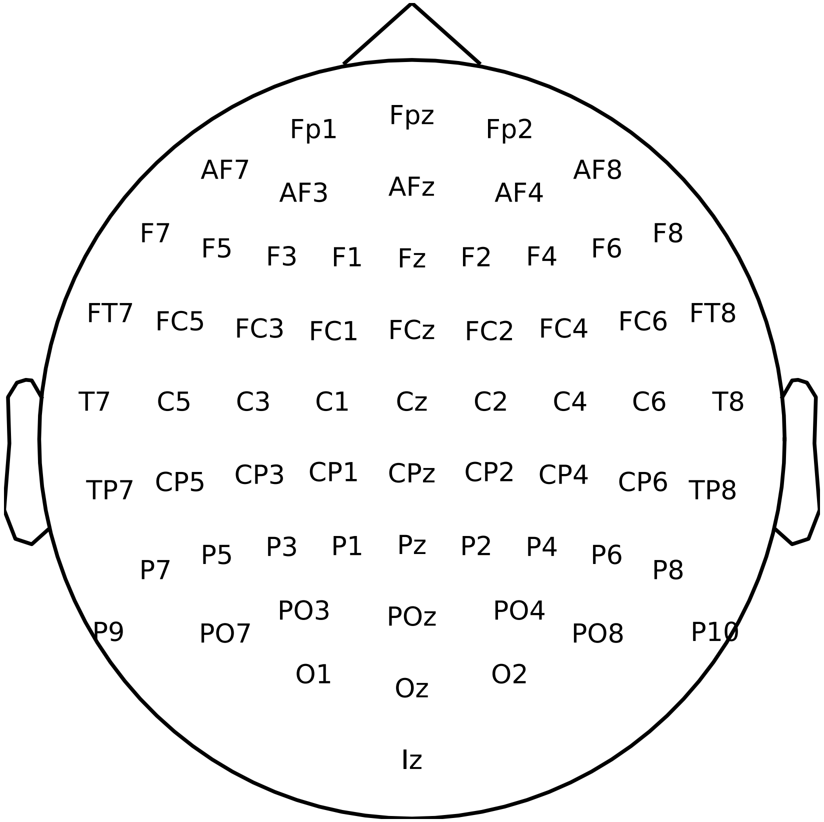
Event-Related Potentials (ERPs)

Event-Related Potentials (ERPs)

"Scalp-recorded neural activity that is generated in a given neuroanatomical module when a specific computational operation is performed."

Luck (2005). *An Introduction to the Event-Related Potential Technique.*

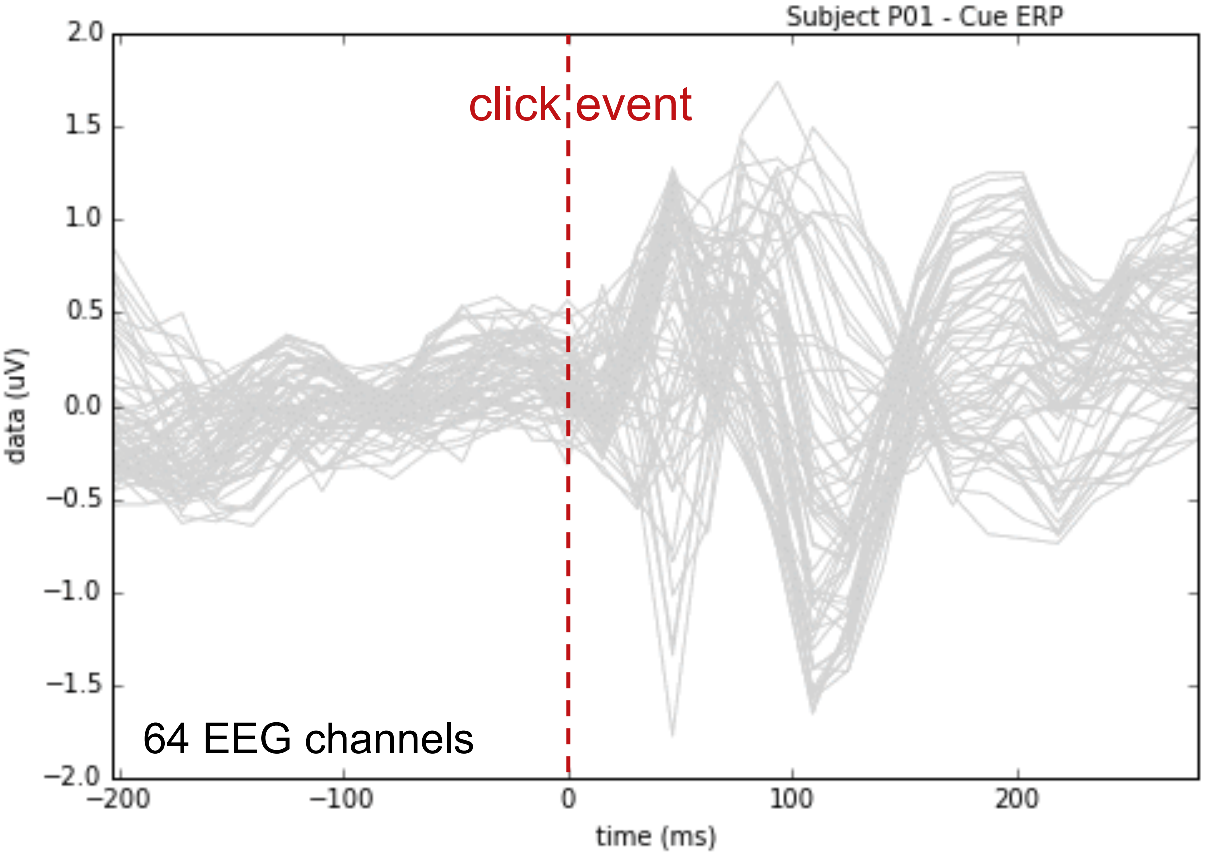
Electroencephalography (EEG)



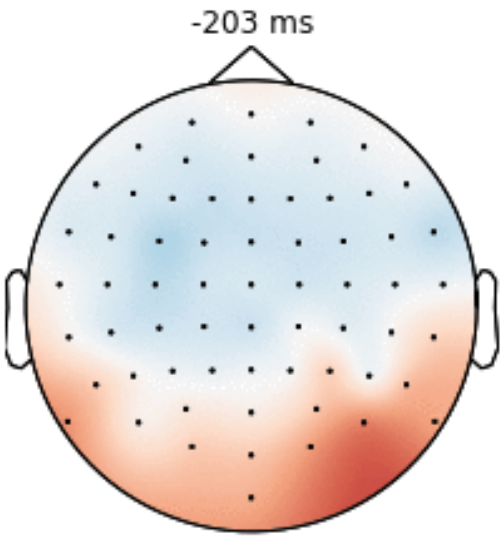
64-electrodes cap (Biosemi)

EEG Visualization

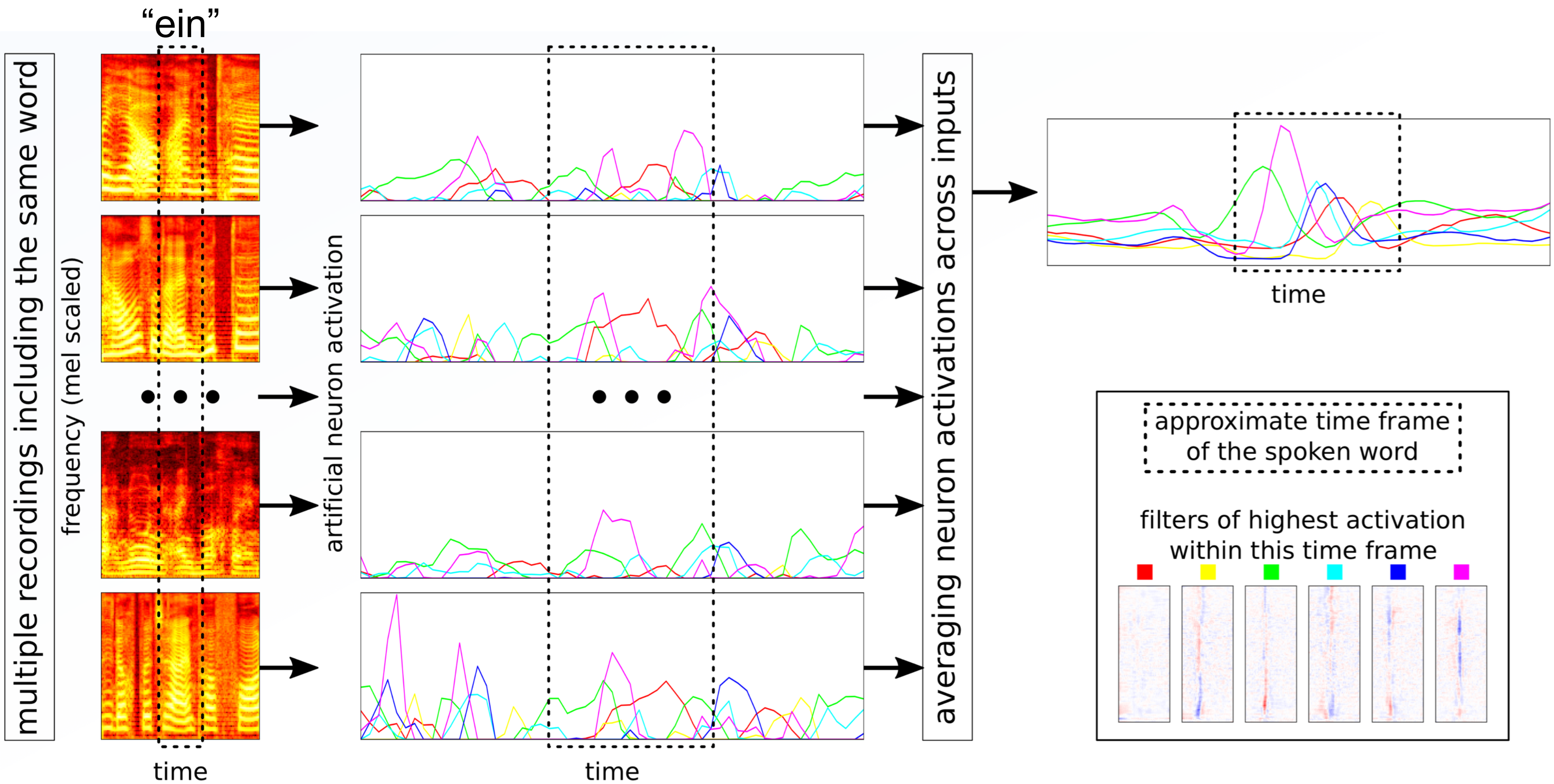
time series
(temporal view)



topographic map
(spatial view)

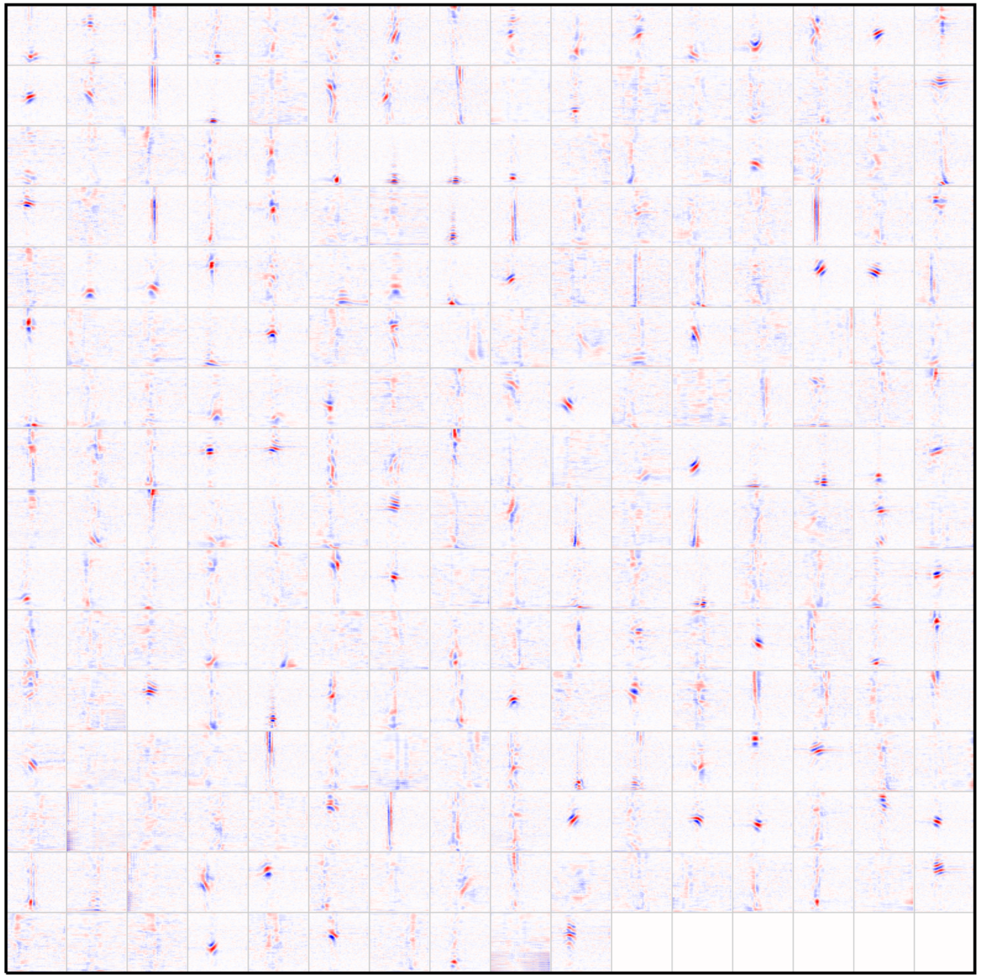


ERP-Like Analysis



- neuron activations are deterministic
- variance lies in the stimuli
  (differences in context, talking speed, pronounciation)

ERP-Like Analysis

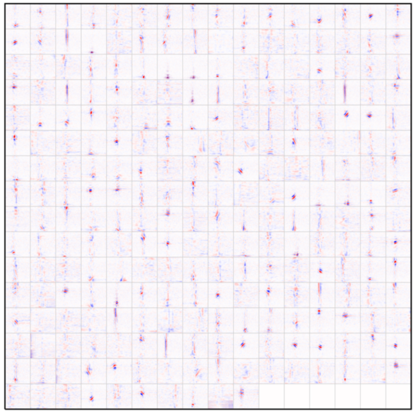


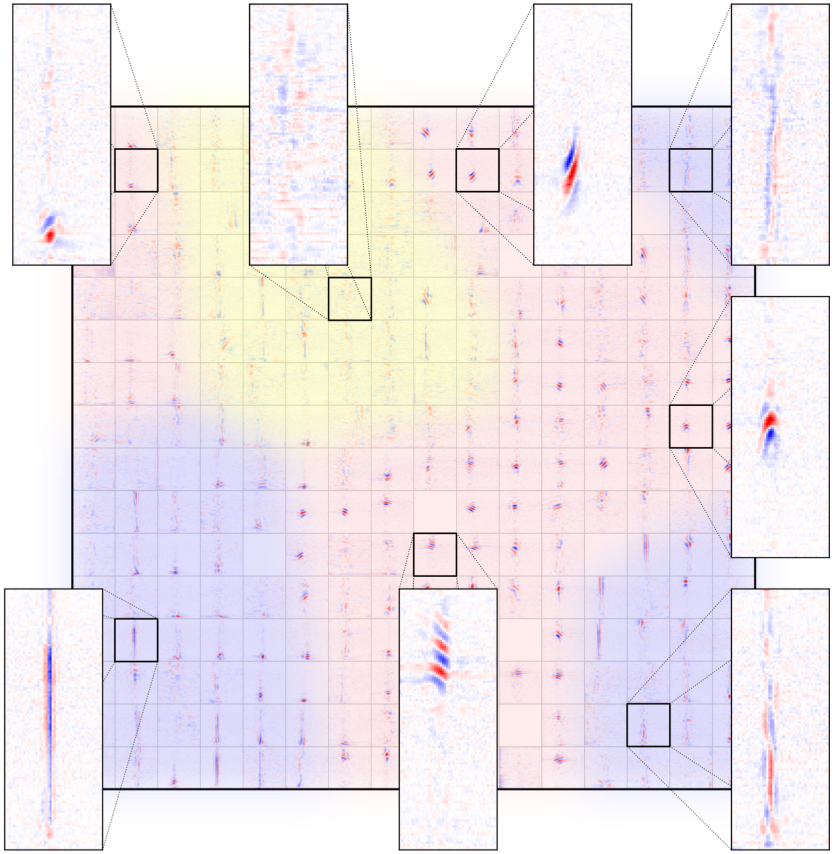
problem:
filters are learned without particular order

250 filters of layer 1

ERP-Like Analysis

re-arrange filters
by similarity using a
self-organizing map

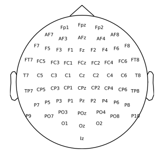


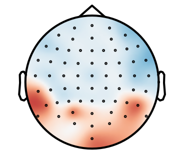


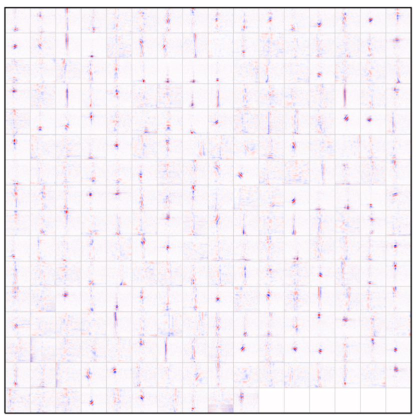
blue areas: beginning or ending of sounds, percussive sounds
red areas: rising and falling pitches in different frequencies
yellow area: noisy sounds

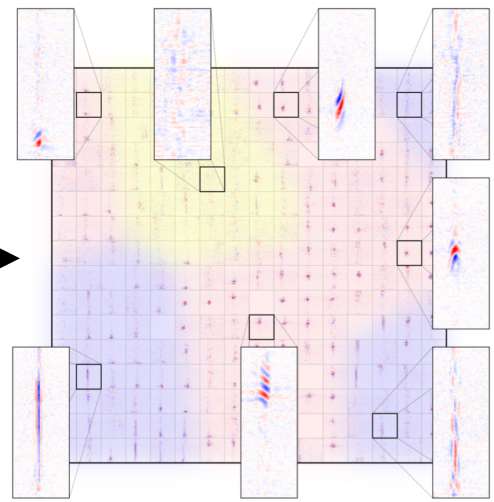
ERP-Like Analysis

EEG equivalent:

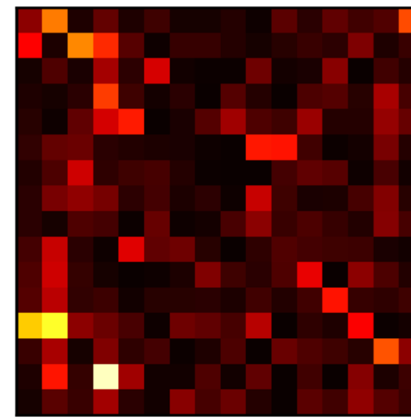






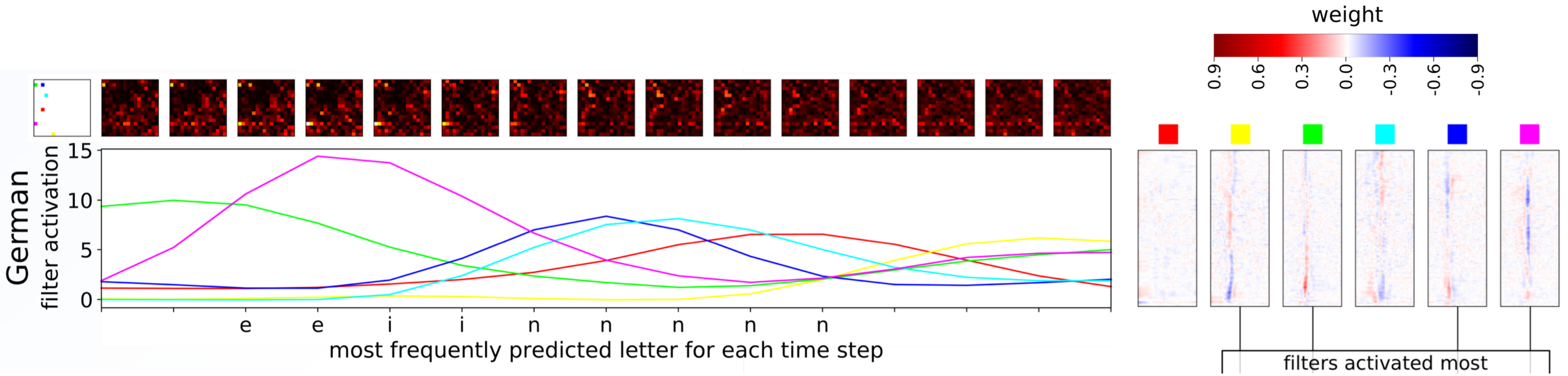


activation map

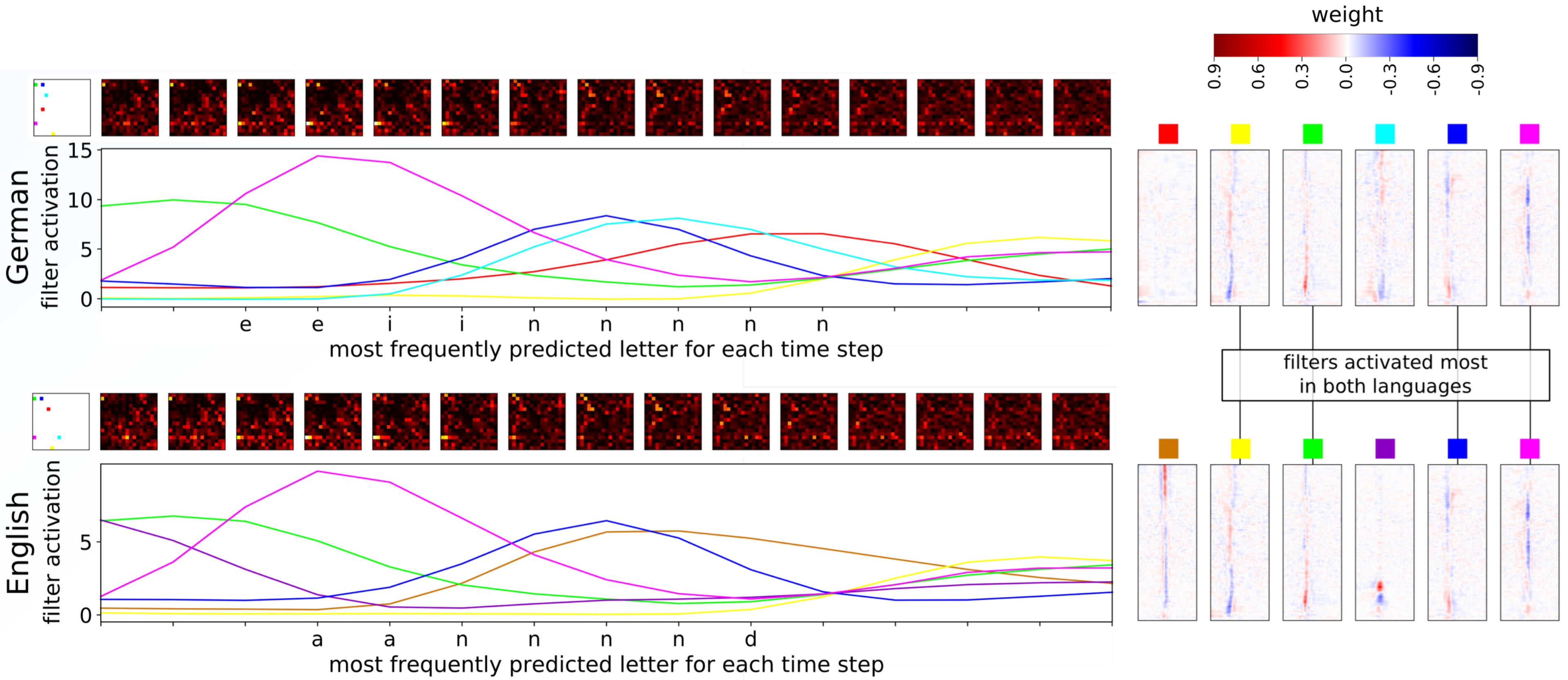


blue areas: beginning or ending of sounds, percussive sounds
red areas: rising and falling pitches in different frequencies
yellow area: noisy sounds

ERP-Like Analysis

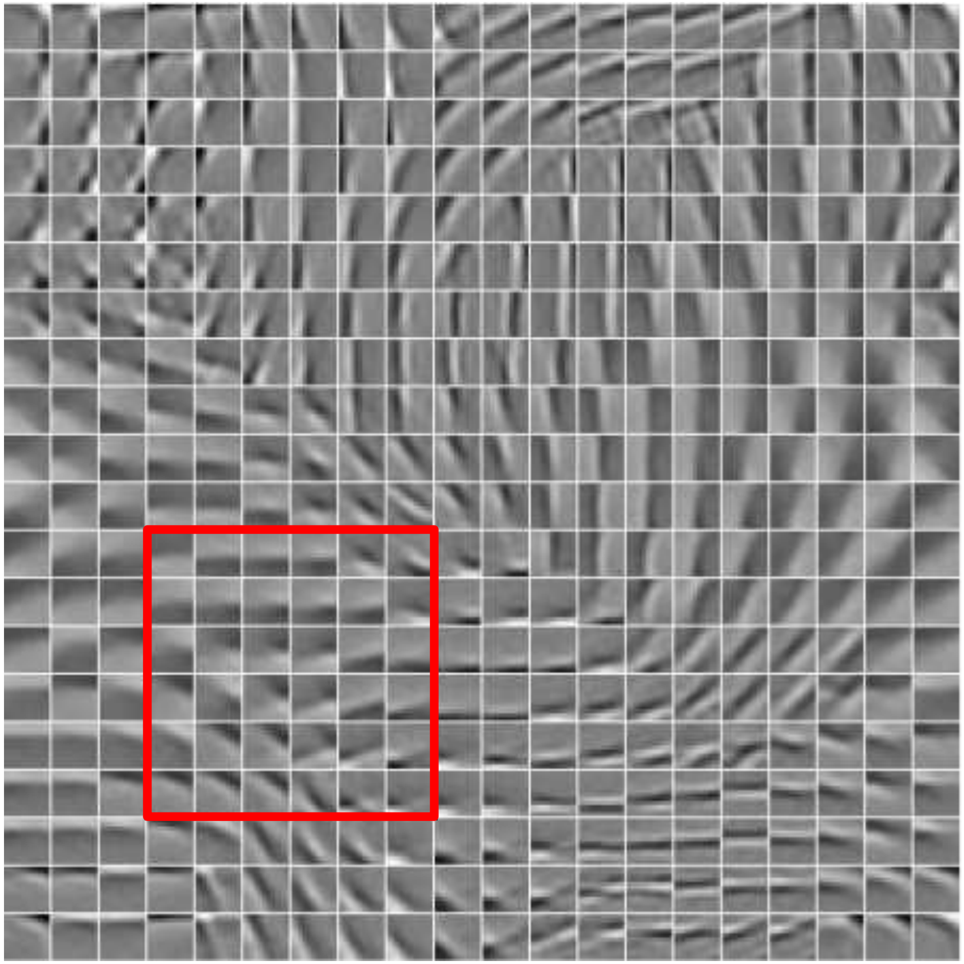


ERP-Like Analysis



- highly similar neuron activations in English and German, but language-specific predictions

Topographic Filter Maps



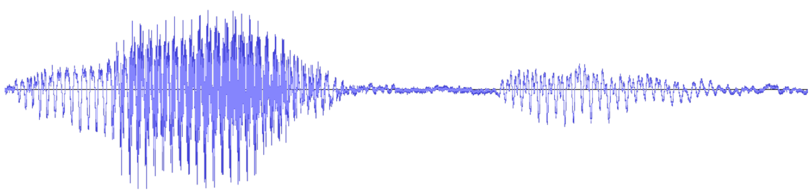
neighborhoods of similar filters

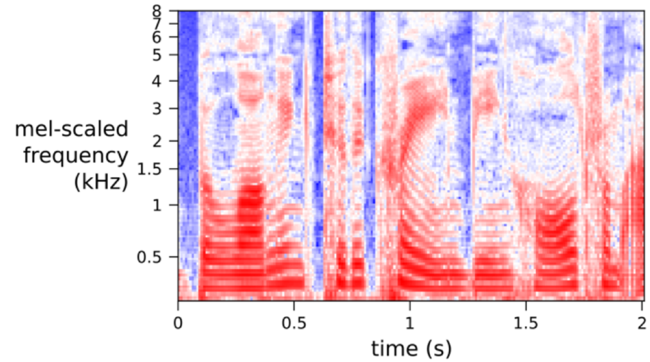
K. Kavukcuoglu, R. Fergus & Y. LeCun.
"Learning invariant features through topographic filter maps."
Computer Vision and Pattern Recognition, 2009. CVPR 2009.

Deeper Analysis: Neuron Activation Profiles (NAPs)

Introspection for Audio Data

- We have
  … little intuition about input signal





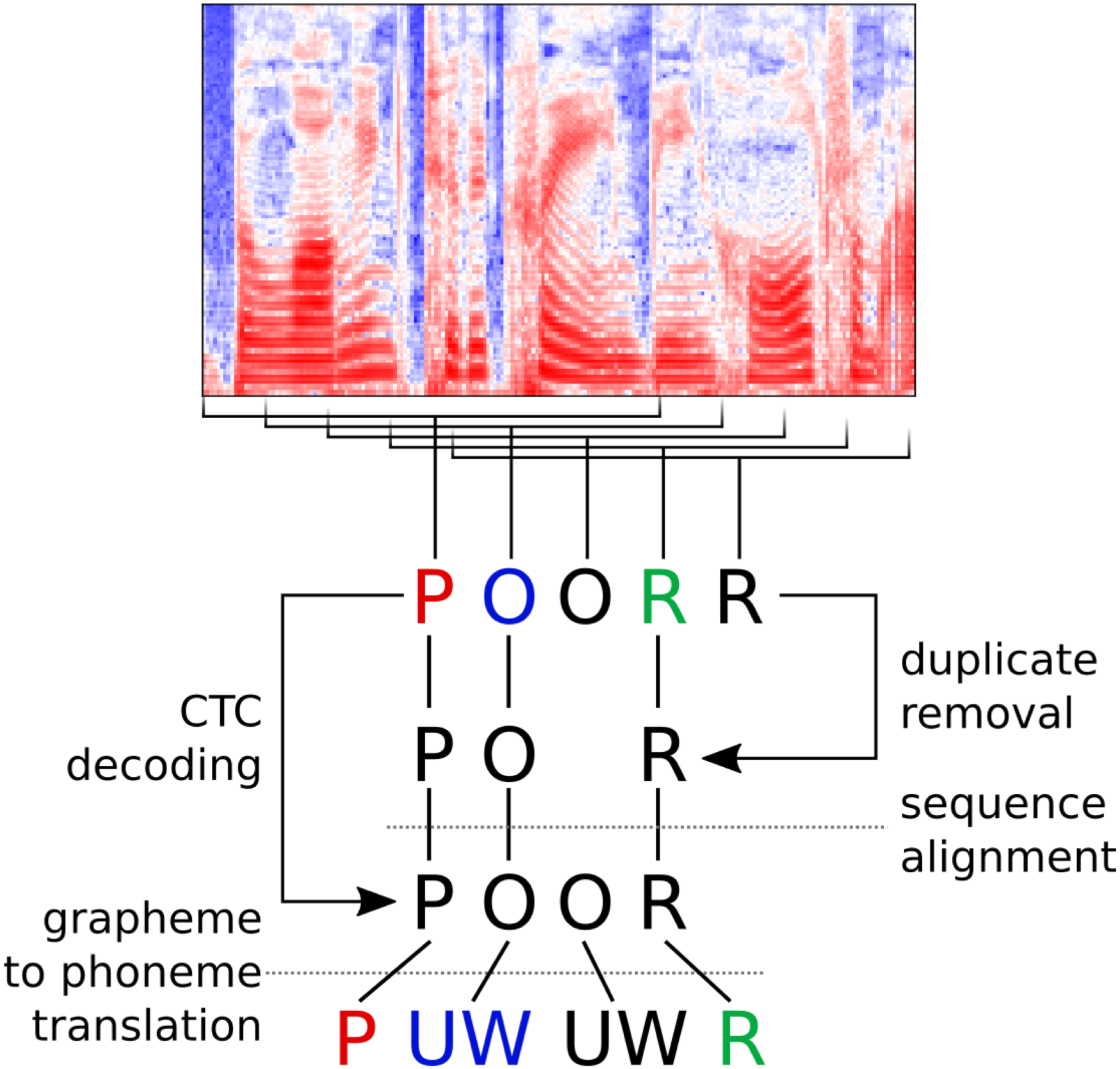
… more intuition about the output

`‘SPEECH’ → /S P IY CH/`

Introspection for Audio Data

- Instead of saliency maps or activation maximization:
  – obtain layer-wise class-specific network responses
  – compare their similarities to human intuition

Deriving Phoneme Annotations



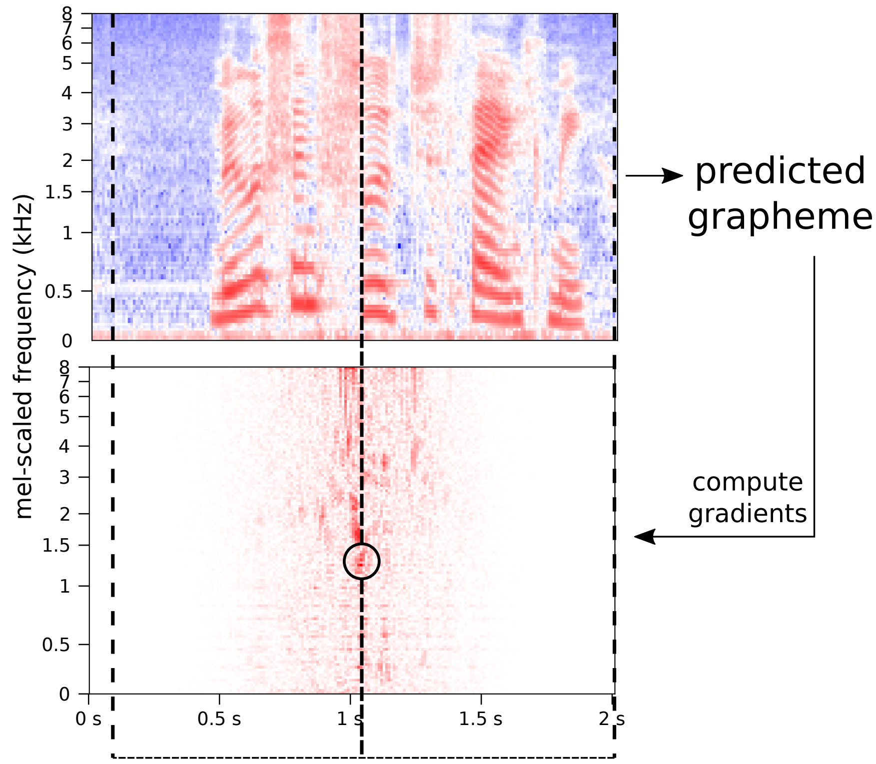
Needleman & Wunsch: “A general method applicable to the search for similarities in the amino acid sequence of two proteins.” *Journal of molecular biology*, 48(3):443–453, 1970.

attention-based encoder-decoder
encoder: 2 bi-LSTM layers
decoder: global attention + 2 LSTM layers
trained on CMU Pronunciation Dictionary

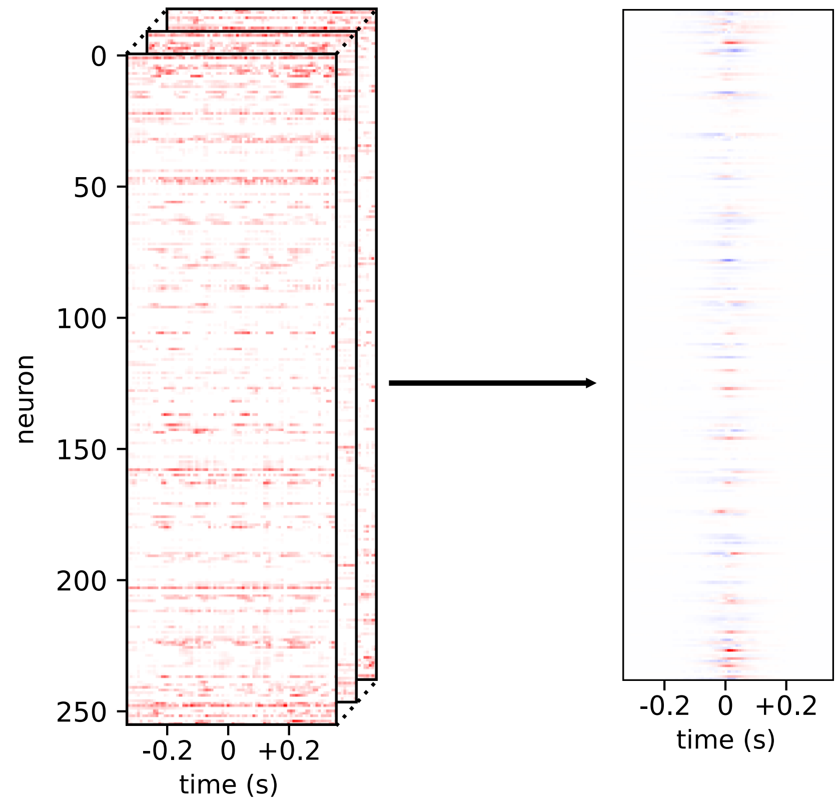
Krug, Knaebel & Stober: “Neuron Activation Profiles for Interpreting Convolutional Speech Recognition Models“ In: IRASL Workshop @ NeurIPS 2018.

Characteristic Network Responses

center at highest importance:
$\mathrm{argmax}_t(|\mathrm{gradient}| \odot \mathrm{activation})$

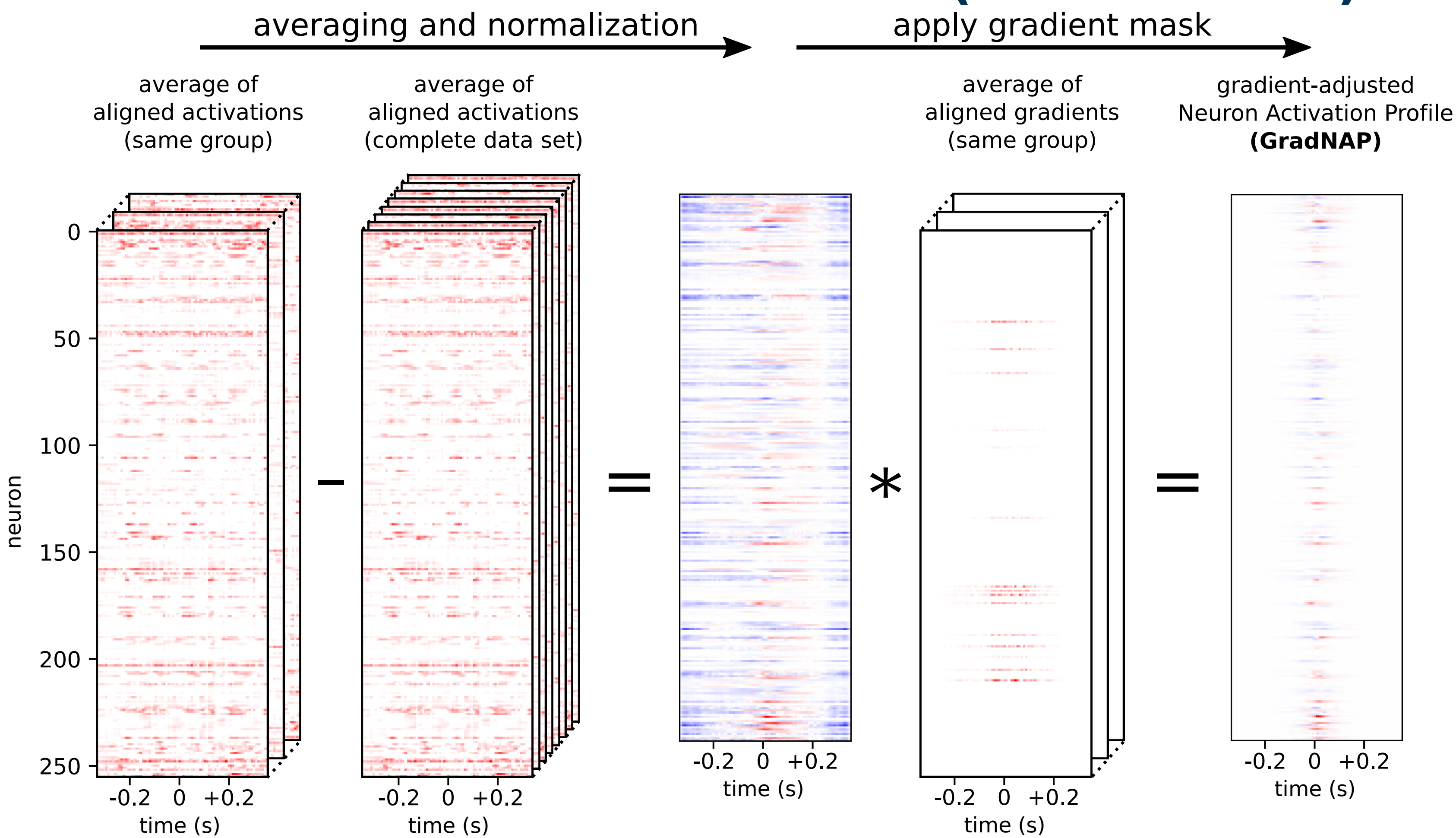


speech alignment



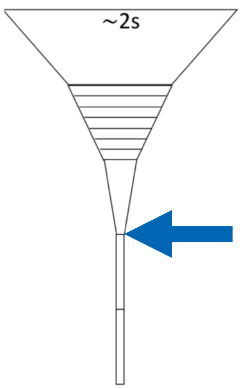
response averaging

Gradient-adjusted Neuron Activation Profiles (GradNAPs)



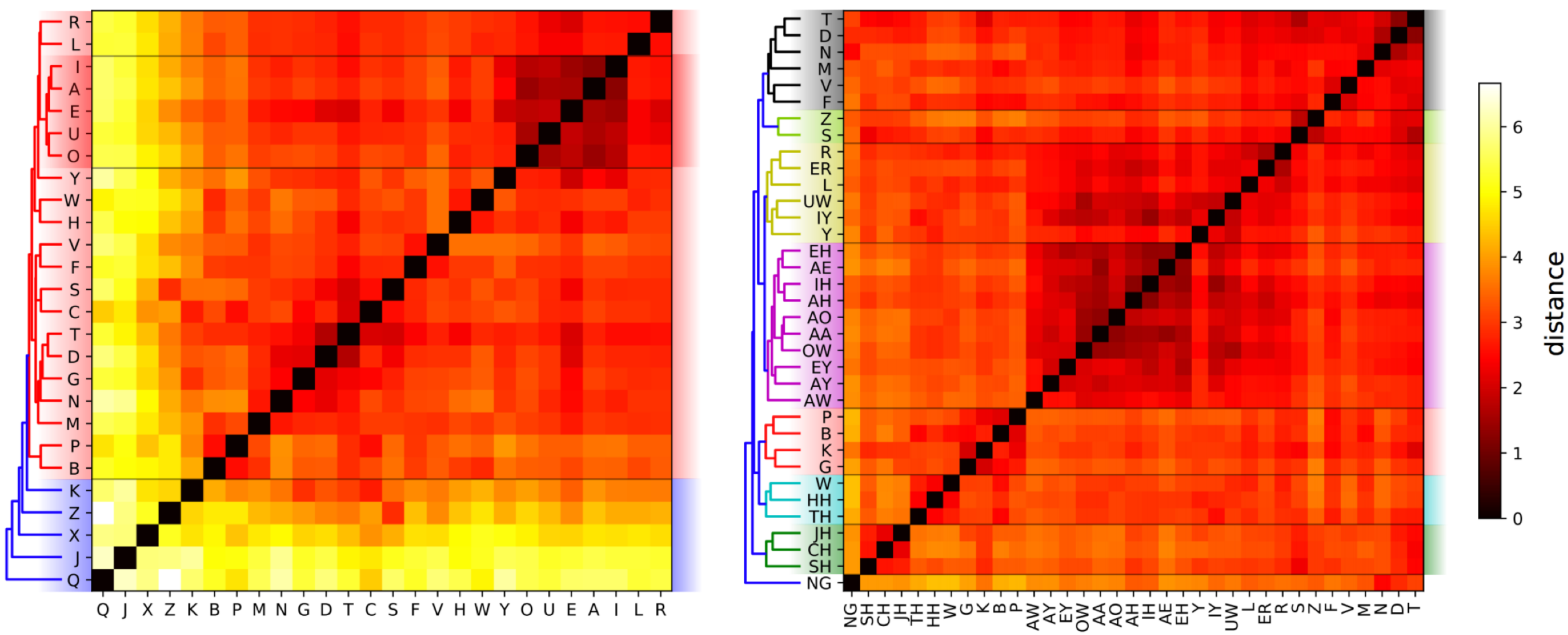
- use sensitivity-based alignment
- use sensitivity values to mask out activations of low relevance for prediction

Clustering of NAPs in 9th Layer



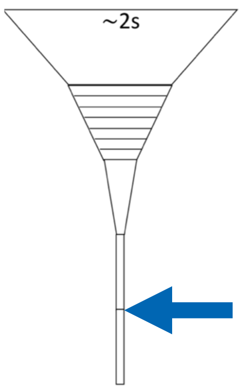
letters

phonemes

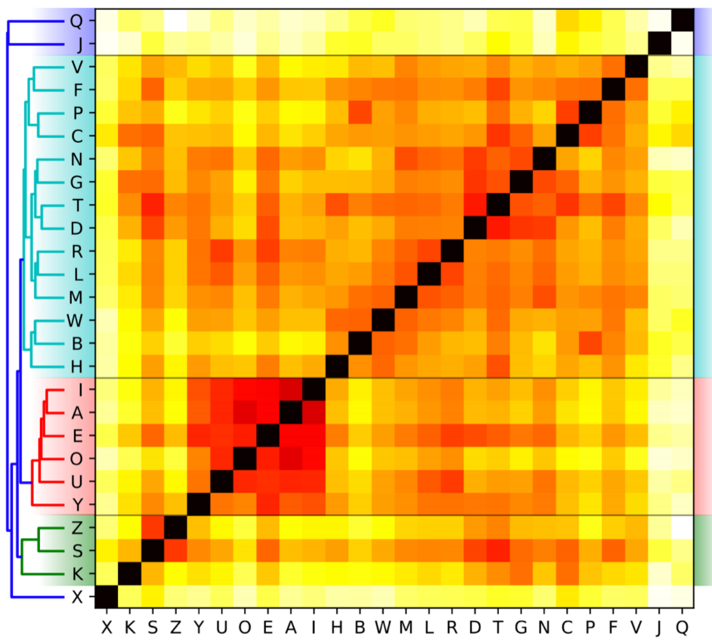


- clusters of similar phonemes emerge
- no distinct clustering of NAPs for letters

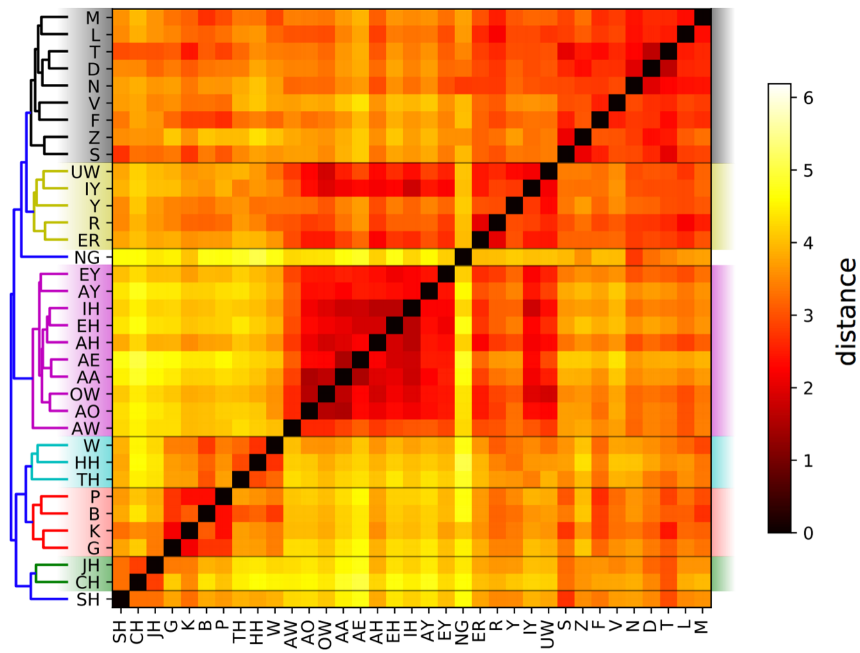
Clustering of NAPs in 10th Layer



letters



phonemes

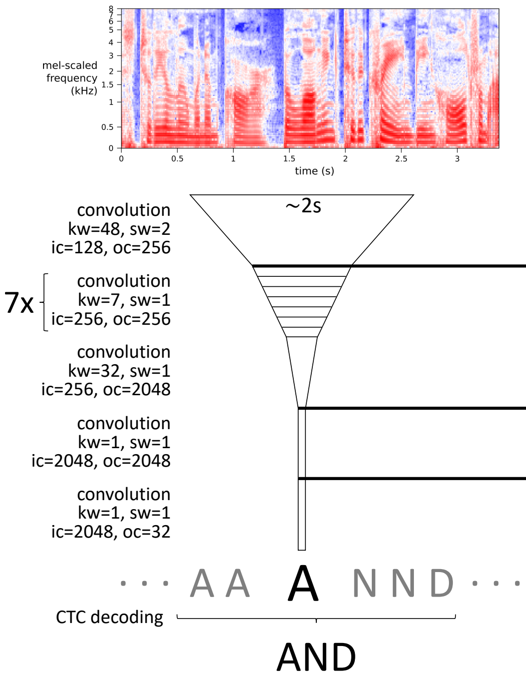


- phoneme clusters become more distinct
  - cluster of vowel letters emerges

Clustering of NAPs

convolutional speech recognition

architecture based on Wav2Letter [Collobert et al, 2016]

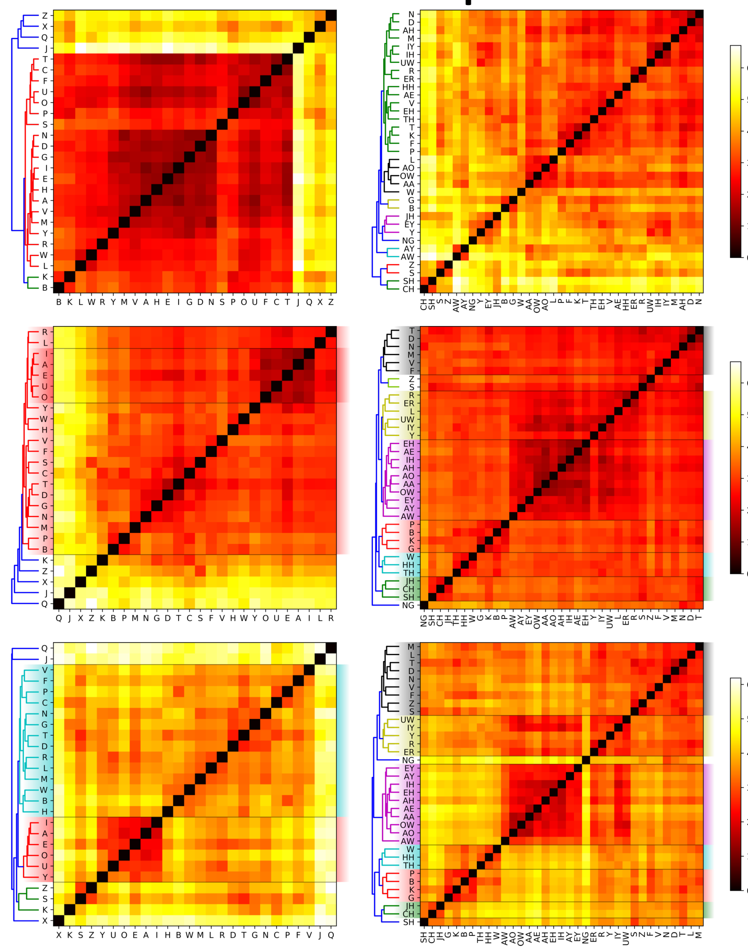


specific problems:

- (1D-)convolutional architecture
- exact time of predicted letter in spectrogram cannot be determined

clustering of Neuron Activation Profiles

letters **phonemes**



- no clusters of similar phonemes or letters
- layer only detects basic acoustic features

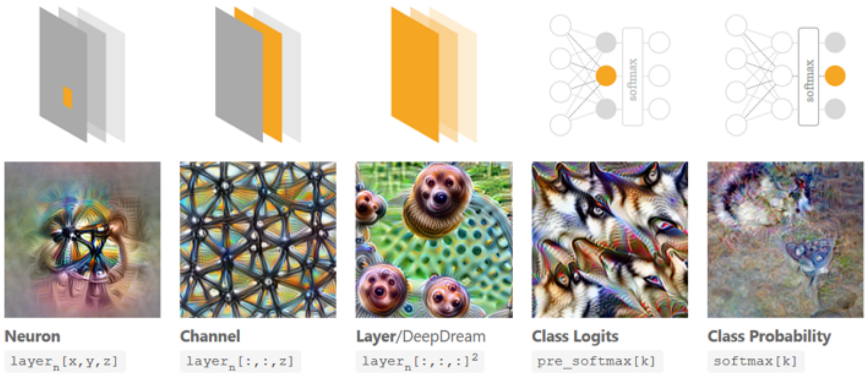
- clusters of similar phonemes emerge
- no distinct cluster of similar letters

- clusters of similar phonemes become more distinct
- clusters of vowel and consonant letters emerge

Krug, Knaebel & Stober: “Neuron Activation Profiles for Interpreting Convolutional Speech Recognition Models“ In: IRASL Workshop @ NeurIPS 2018.

Recap: Introspection

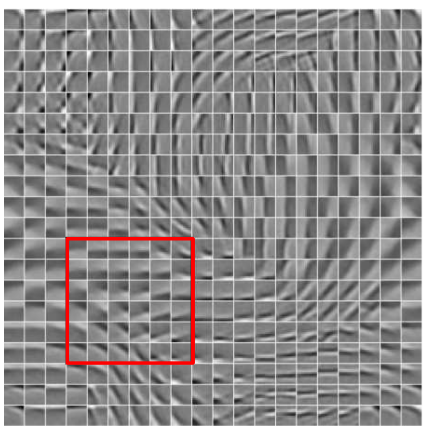
feature visualization (optimize input)



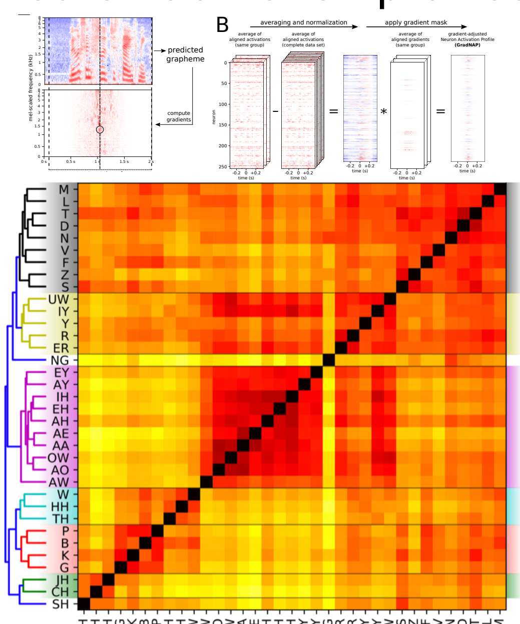
relevance / saliency analysis (for given input)

(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat'

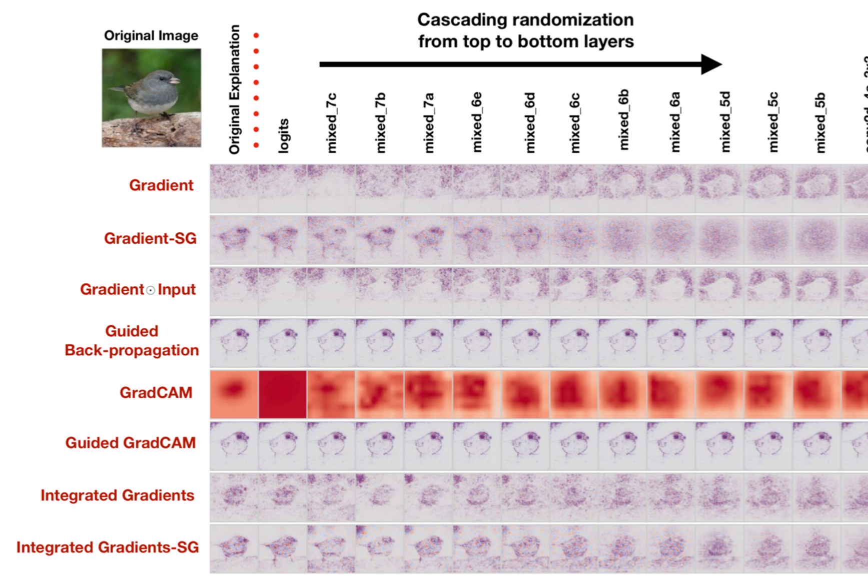
feature topography
(improve interpretability)



neuron activation profiles



=> sanity checks!



Hands-on: Distill / Lucid Tutorials

- Start at
  https://distill.pub/2017/feature-visualization/

- All images were generated using Lucid
  https://github.com/tensorflow/lucid
  (Scroll down for a list of notebooks!)