

ASR, ..., transformers, ... explainability

Louis ten Bosch
Nijmegen/Graz 2025



Short bio

- Mathematics in Leiden, NL
- PhD at University of Amsterdam, Louis Pols
 - Structure of vowel systems
 - Phonetics - phonology
- Postdoc at Institute of Perception Research, Eindhoven
 - Intonation
- ASR department of a Belgium-based speech technology company
- Since 2002: Radboud University, Nijmegen, NL
 - Computational models of human speech processing
 - Computational cognition (learning, language acquisition)
 - Phonetics, morphology
 - Explainable AI, InDeep

INDEEP

INTERPRETING DEEP LEARNING MODELS

FOR TEXT AND SOUND



nationale
wetenschaps
agenda



Radboud University



AI text and image analyses

- **Artificial Intelligence for Text and Image Analysis**
 - Using AI tools (e.g., ChatGPT, GitHub Copilot, open-source LLMs) for building and analyzing text corpora.
 - Applying AI for **text and image classification** and **validation** in research.
 - Emphasis on **ethical considerations** in AI-driven studies.
- **Large Language Models (LLMs) & ChatGPT Applications**
 - Employing LLMs for **data extraction** and **structuring unstructured text** in the social and health sciences.
 - Understanding the **role of ChatGPT** in research workflows.
- **Generative AI for Research Enhancement**
 - Integrating **generative AI** for **data analysis**, **automation**, and **visualization**.

**Shoebox 2,
1961/2,
IBM**



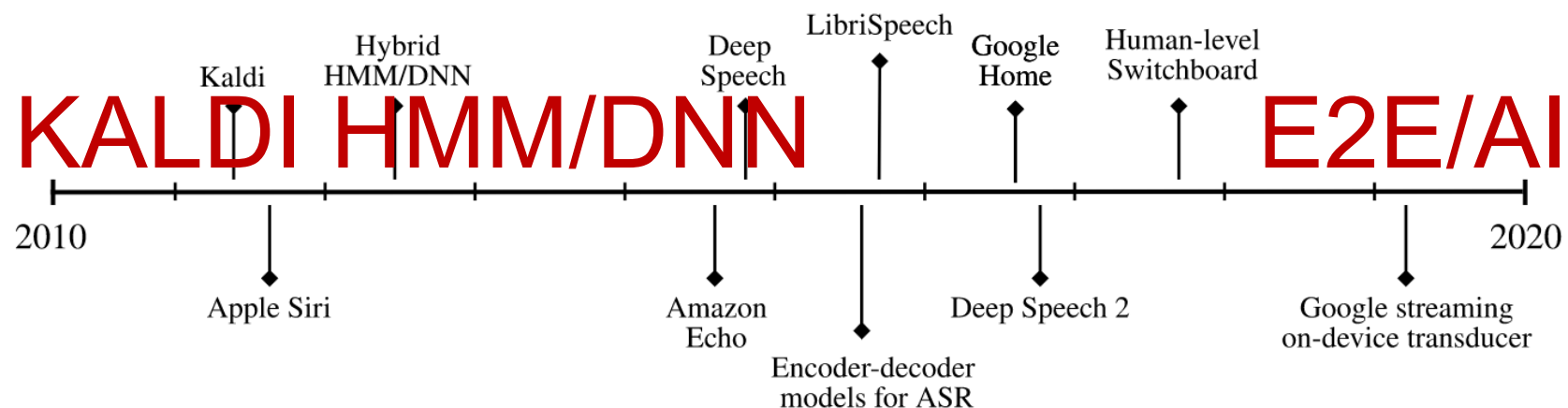
Four ways to look at ASR

- ASR as a **research topic itself**
 - Word error rate (WER) minimization, latency, robustness, network architectures (KALDI vs Whisper), training regimes, transfer learning, etc.
- ASR as **black box module** in larger system
 - In a dialogue system
 - As a tool for the hearing-impaired
 - In cases where typing is dangerous/impossible
- ASR as a **tool** to unravel the structure in the speech signal
 - Biomarking
 - 'Computational lens'
 - CAPT etc.
- ASR as **proxy** for computational simulation for **human speech processing**
 - Plausibility issues, ecology

Key applications of ASR

- **Video:** Real-time and asynchronous video captioning
- **Audio** annotation (verbatim, summarizing), (speaker diarization)
- **Meeting** transcriptions
- **Media monitoring:** Speech-to-Text can help broadcast TV, podcasts, radio, and more quickly and accurately detect brand keyword spotting
- **Command and control** (robots)
- **Telephony:** Call tracking, accurate transcriptions, call analytics, speaker diarization, and more.
- **E-health:** Medical analyses of patients' audio (COPD, Alzheimer, Parkinson etc.). **Speech as biomarker. Verbatim or corrected output??**

ASR 2010-now

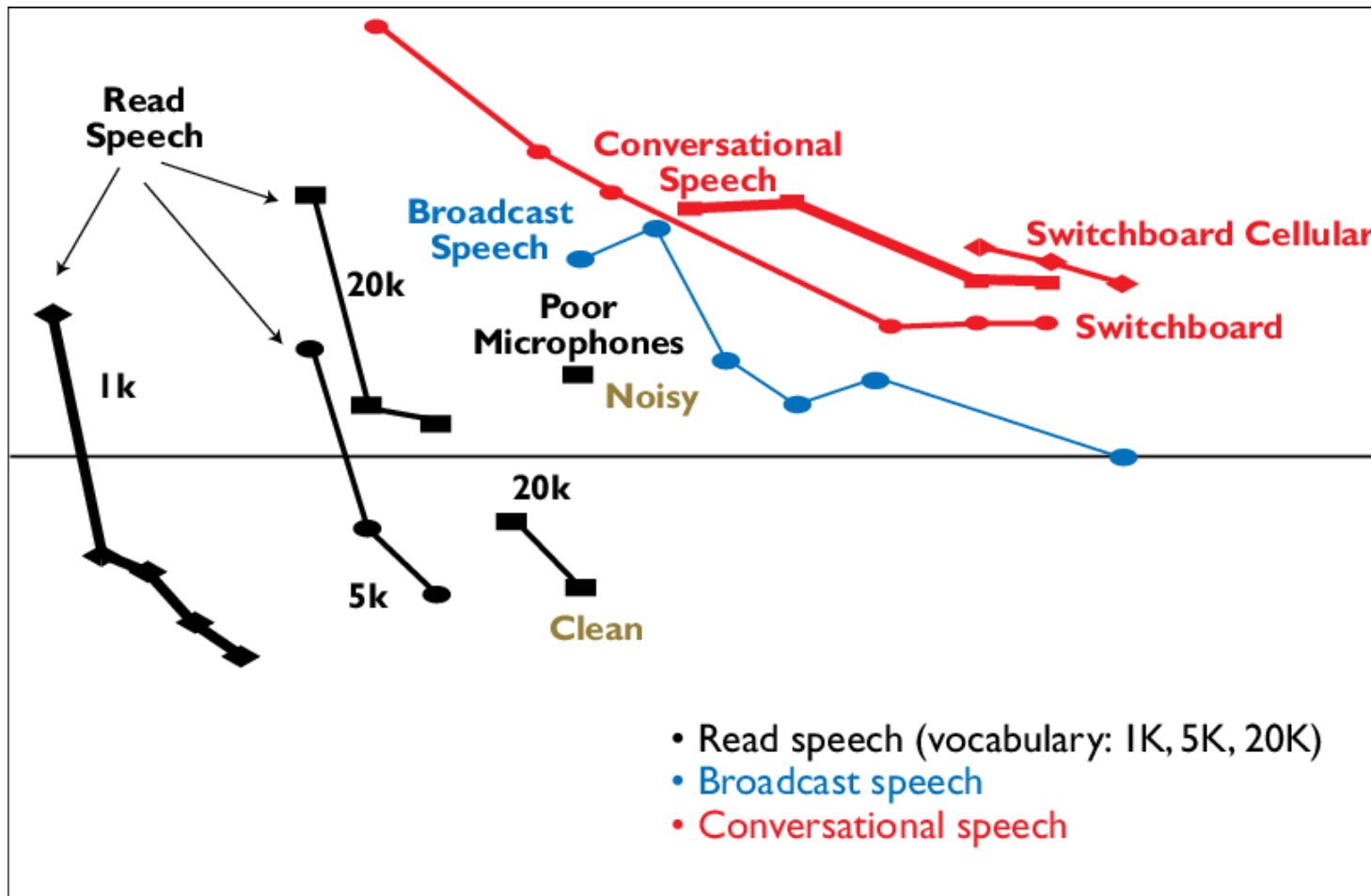


A timeline of some of the major developments in speech recognition from the years 2010 to 2020. The decade saw the launch of voice-based devices and voice assistants, open-source and widely used speech recognition software like Kaldi, and larger benchmarks like LibriSpeech. We also saw speech recognition models improve starting from hybrid neural network architectures to more end-to-end models including Deep Speech, Deep Speech 2, encoder-decoder models with attention, and transducer-based speech recognition.

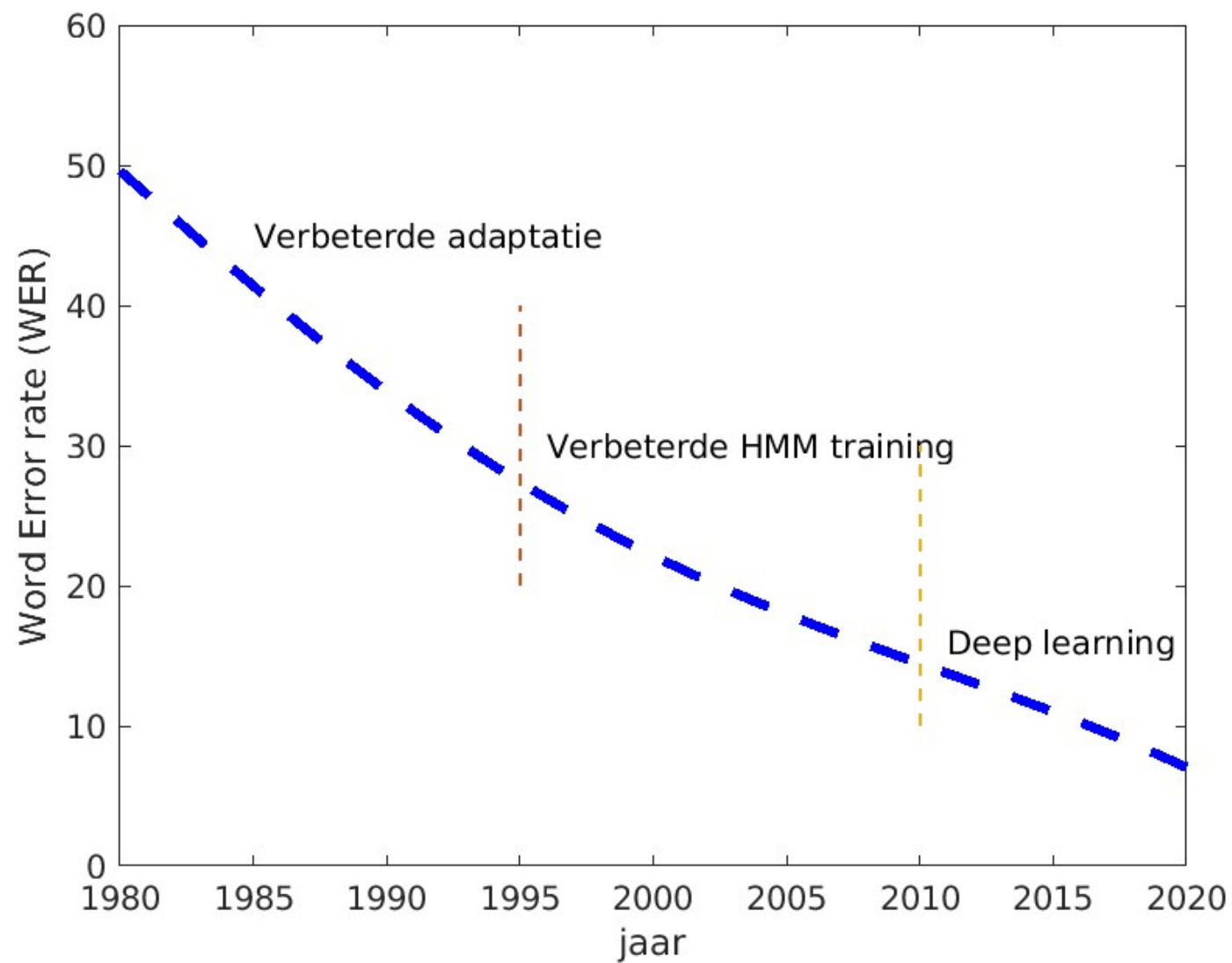
[From https://awni.github.io/future-speech/](https://awni.github.io/future-speech/)

ASR techniques, overview

- **Template matching ('70)**
 - Representative speech patterns (e.g. words, syllables) are stored and labeled
 - New unknown data identified by comparison
- **Probabilistic matching ('80-now)**
 - Often by Hidden Markov Modeling (HMM)
 - An HMM is characterized by hidden states graph + state-state transition probabilities
 - Characterized by a (statistical) distribution in the acoustic space
 - Knowledge based
- **ANN-based (2013-now)**
 - Many directions
 - Mapping oriented: audio → orthography
 - Data based
- **AI, Deep Learning (2020-now)**

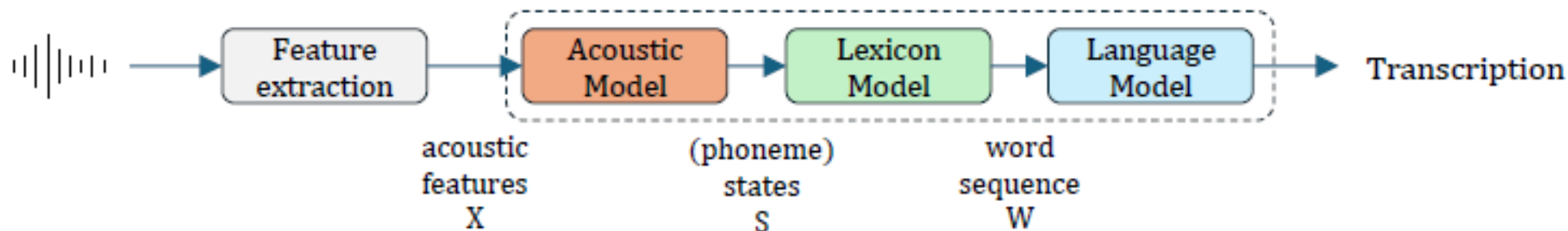


(By Xuedong Huang, 2016)



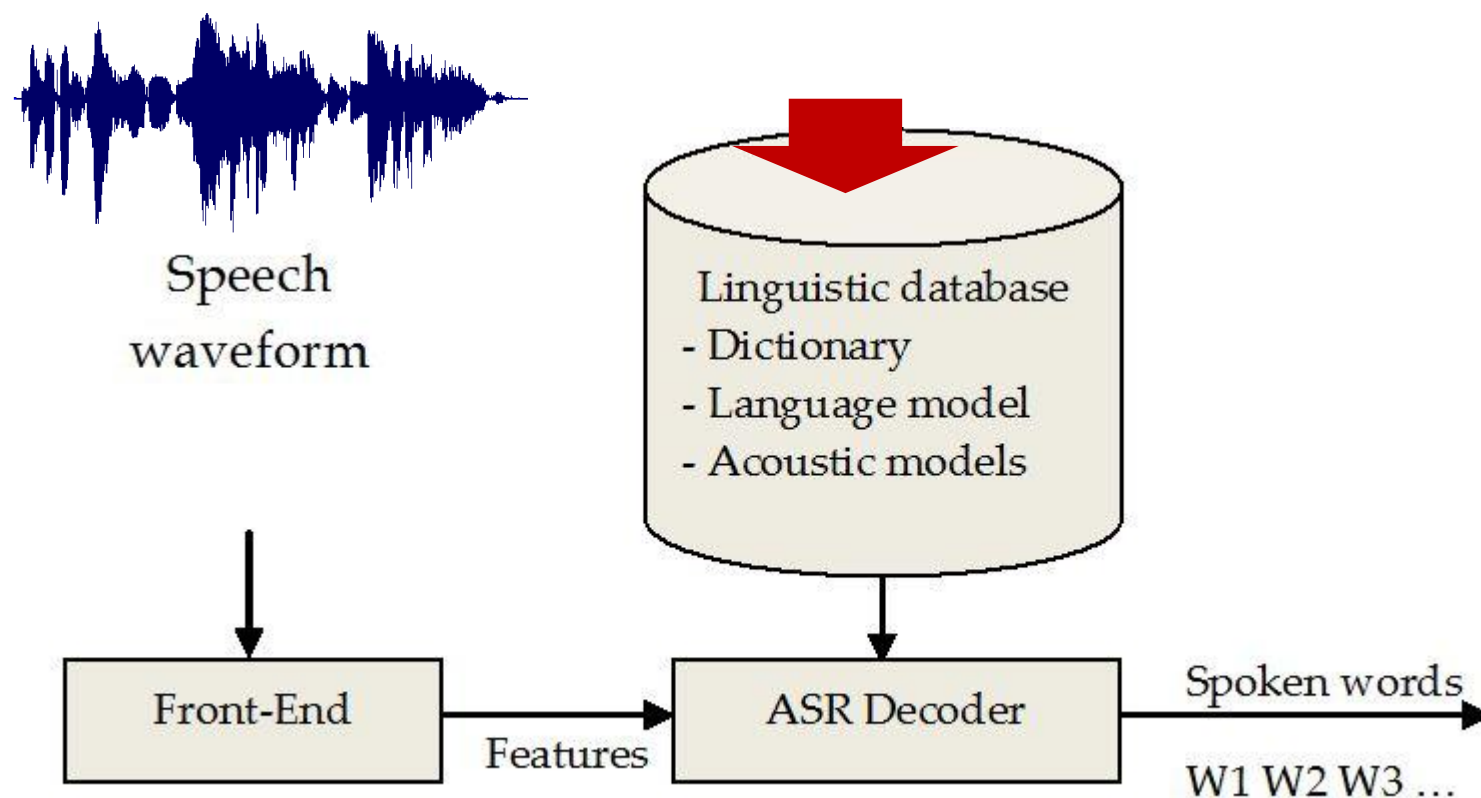
Overview

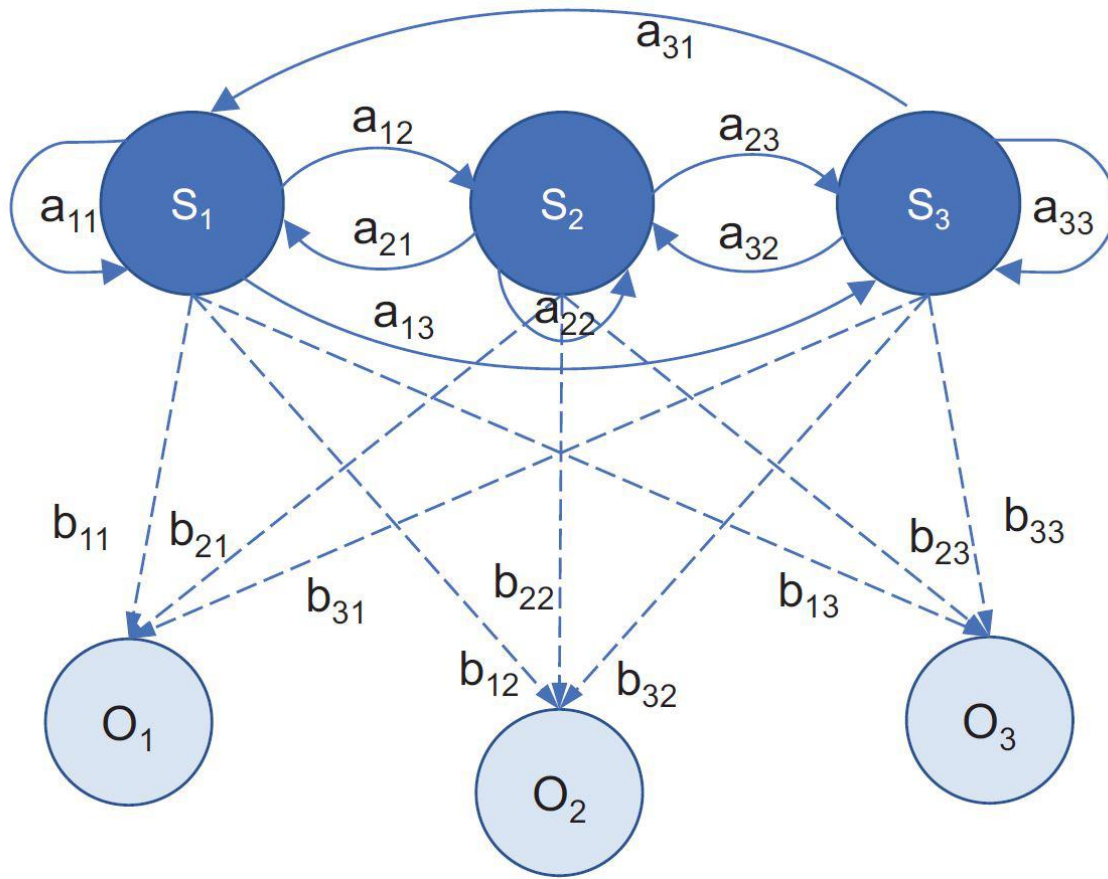
- 1952 “Audrey” (Bell Labs), numbers 0- 9, reported acc 97% , single speaker
- 1961-2: Shoebox (IBM), 16 English words on top of digits, single speaker
- 1993-5 (IBM) first commercial speech recognition product for personal dictation
- 1970’s, **hyrid speech recognition** combining acoustic models, pronunciation dictionaries and language models



- Since 2019/20 or so: **end-to-end speech recognition**: Vectorization of everything!
- Today: Transformer-like architectures and large-scale data.

Hybrid ASR: modular architecture



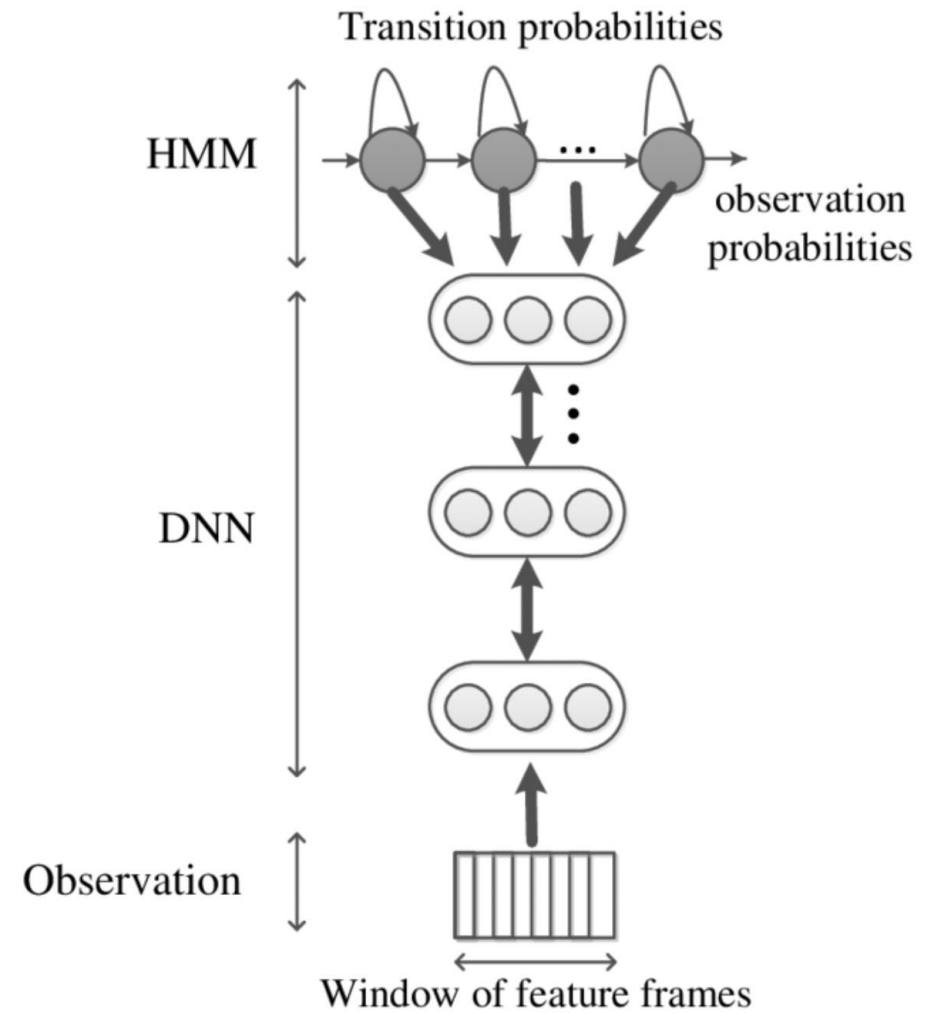
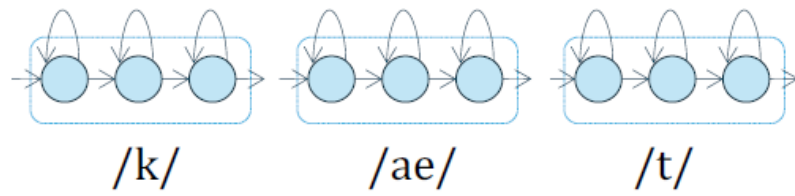


HMM

Forward backward
Expectation Maximization
Viterbi

Pos:
time warping
statistical underpinning

Neg:
Poor time resolution
State-state independence assumption



Deep Learning: a Very Short history ...

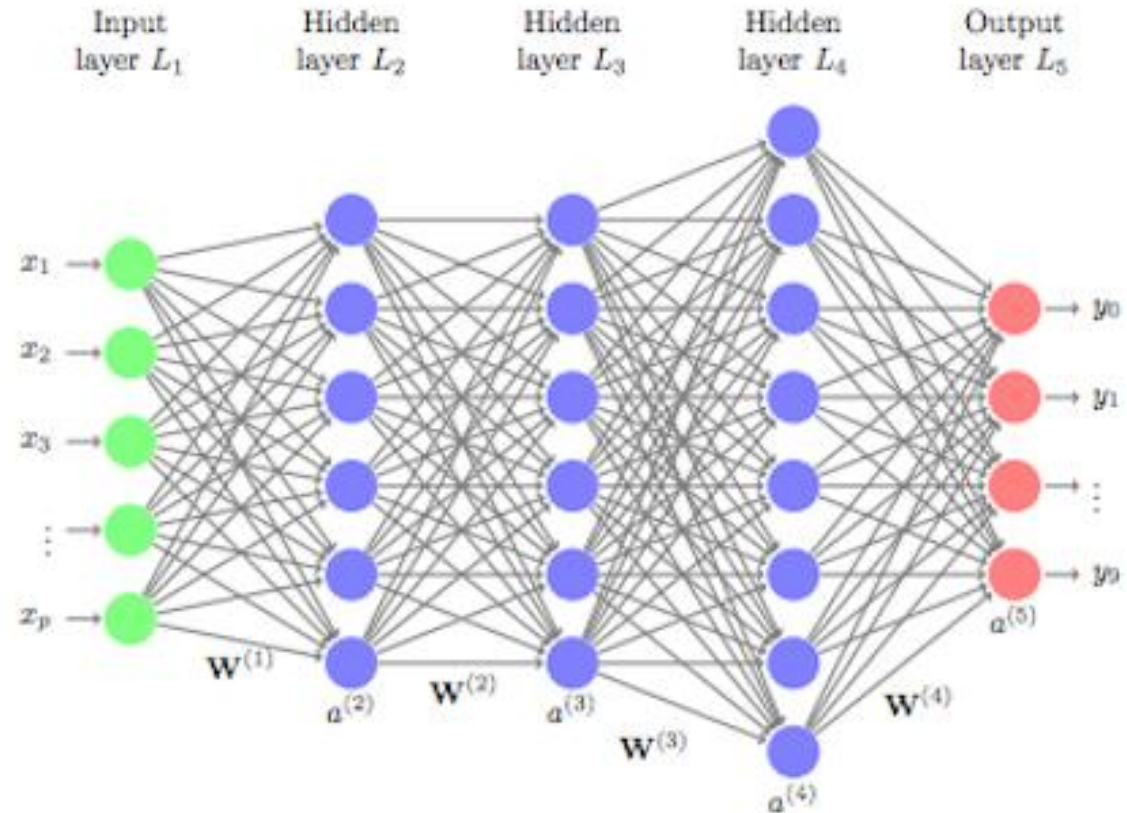
- **MLP** (MultiLayer Perceptron): since 1958 (e.g. Rosenblatt)
- **RNN** (recurrent neural networks): (1925, Ising), 1980, various people (e.g. Elman)
- *Backpropagation: 1982, Werbos a.o.*
- **LSTM** (long-short term memory): 1995, Hochreiter et al.
- *Word2vec: 2013, Mikolov, Google team*
- Sequence to sequence: 2014, Sutskever et al. (now at OpenAI)
- Since 2006: theoretical development of training approaches deep models (Bengio, et al.)
- **Transformers**: 2017, Vaswani et al.
- BERT (Bidirectional Encoder Representations from Transformers), 2018, Google (and many variants)
- **Audio, s2t: Wav2vec2.0, 2020: Whisper, 2022 (and variants), Conformers**

Speech and Language and AI

- Speech (and Language)
Technology now
- Impact of Artificial Intelligence (AI)

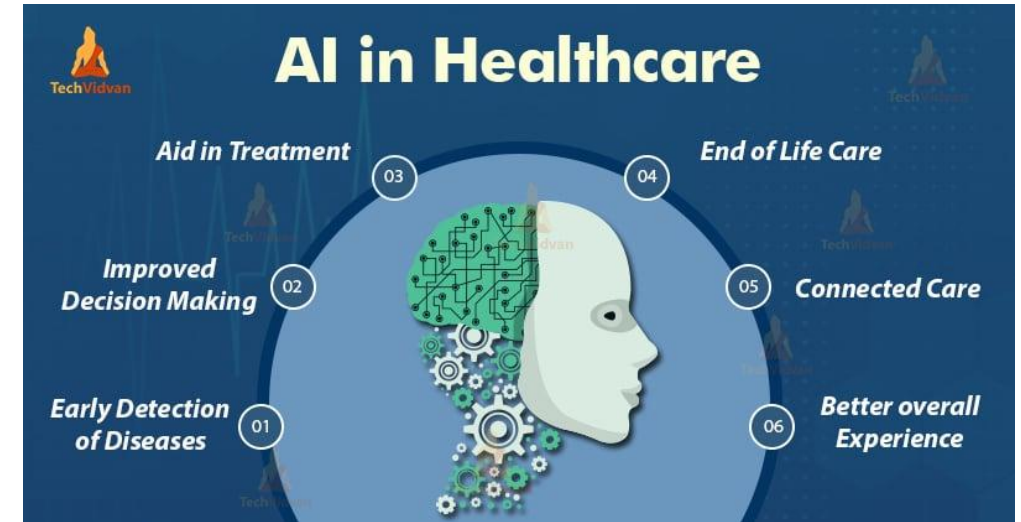
Speech and Language and AI

- Speech (and Language) Technology now
- Impact of Artificial Intelligence (AI)



What we now see is only the beginning...

- AI can be extremely useful:
 - pattern discovery
 - e.g., medical data
 - text generation
 - code generation
- But also risky:
 - spoofing
 - deep fakes, incl. fake audio
 - production of incorrect but convincing texts
 - large-scale pollution of social media eco-system



	pro	challenge
Classical approach	Insight-based, knowledge based	Lower performance
Recent deep learning approaches	Higher performance (relative 30-60% reduction of error rates)	Data hungry!! (But ...!) Small data sets? What do we learn? Explainable, Responsible AI?

One of the KEY papers (2017)

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez*[†]
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin*[‡]
illia.polosukhin@gmail.com

Attention*1706.03762.pdf

Abstract

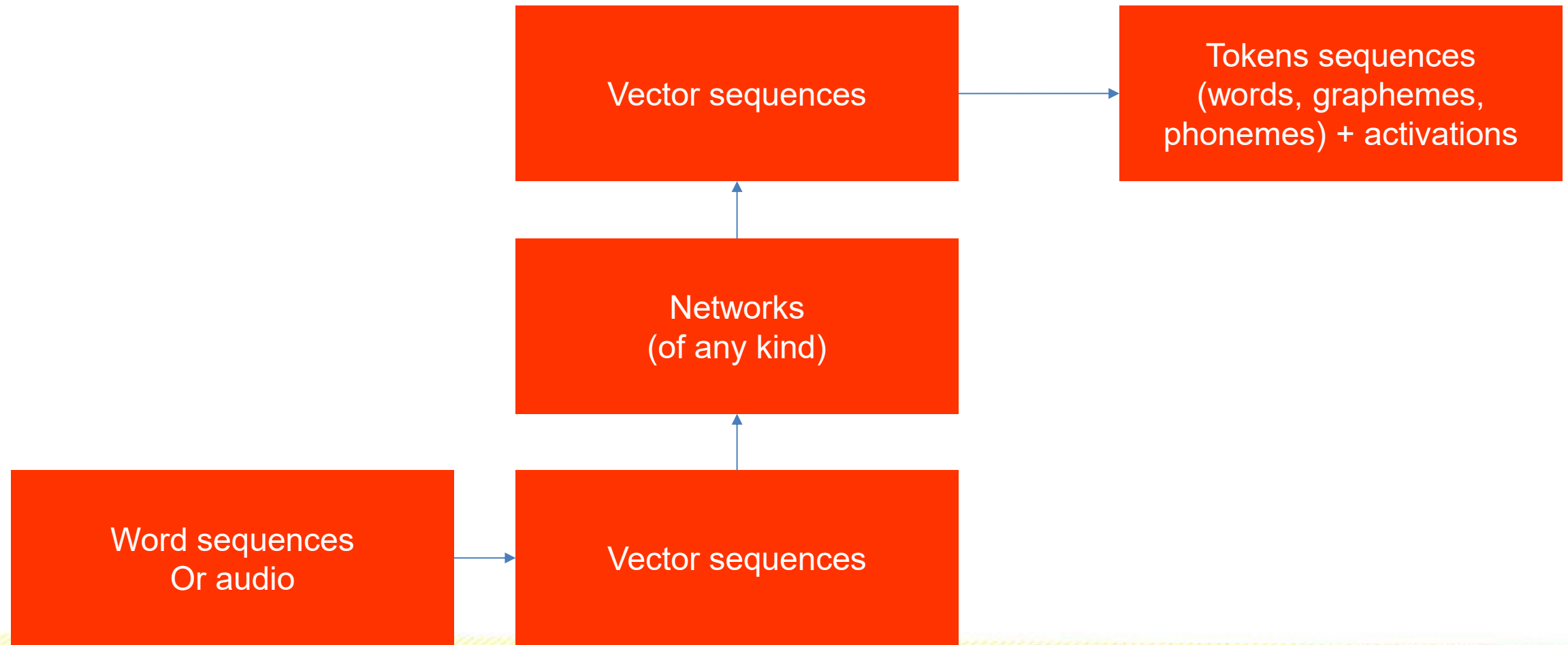
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to

Vaswani et al. (2017)

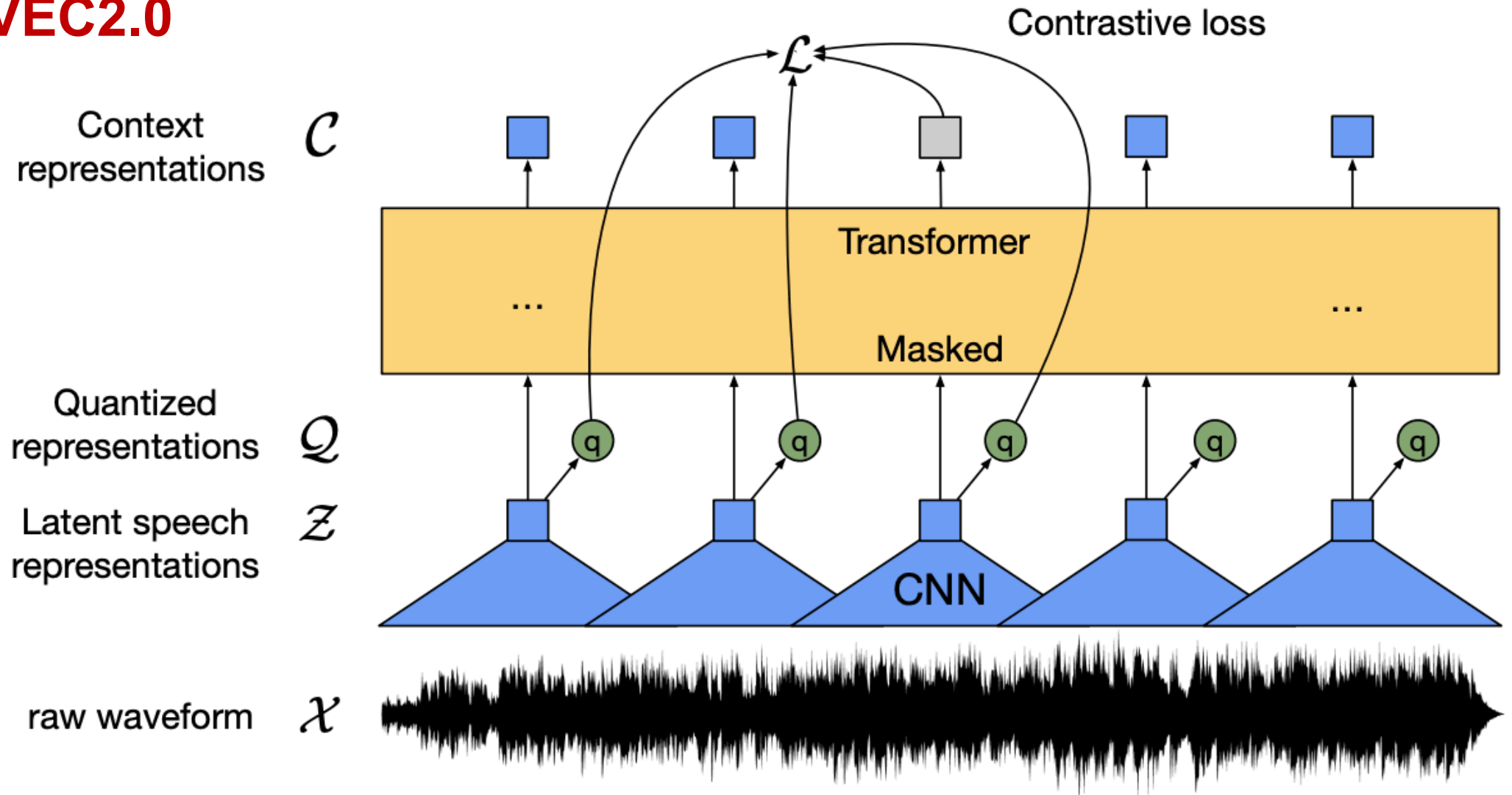
Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

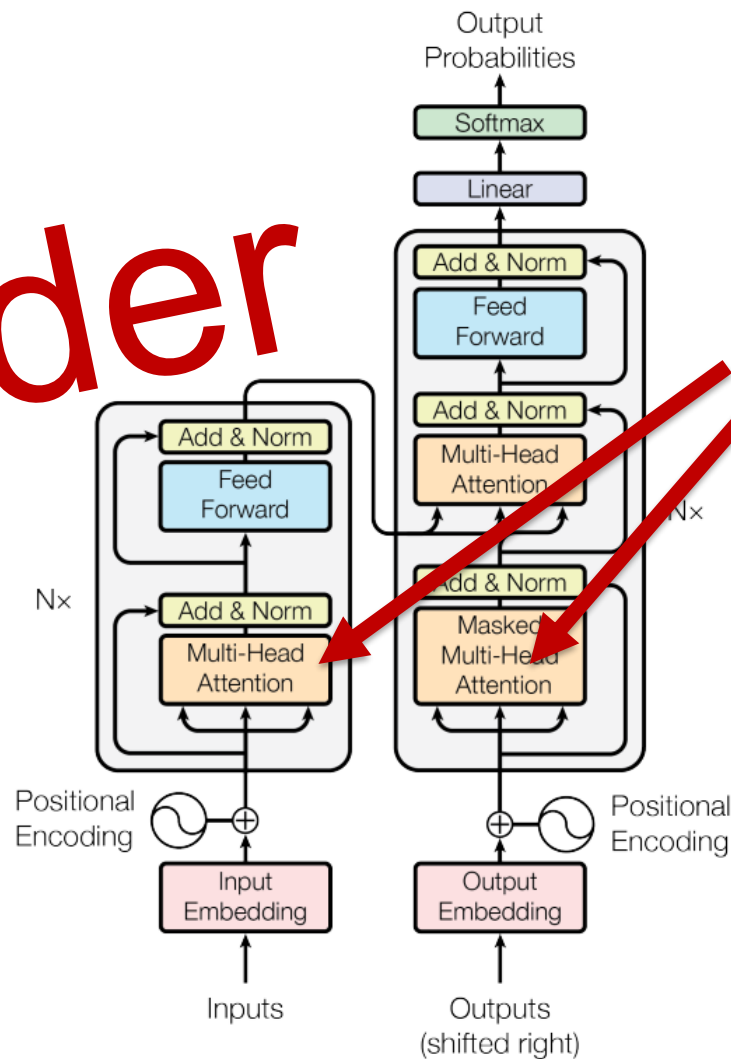
VECTORS, NETWORKS



WAV2VEC2.0



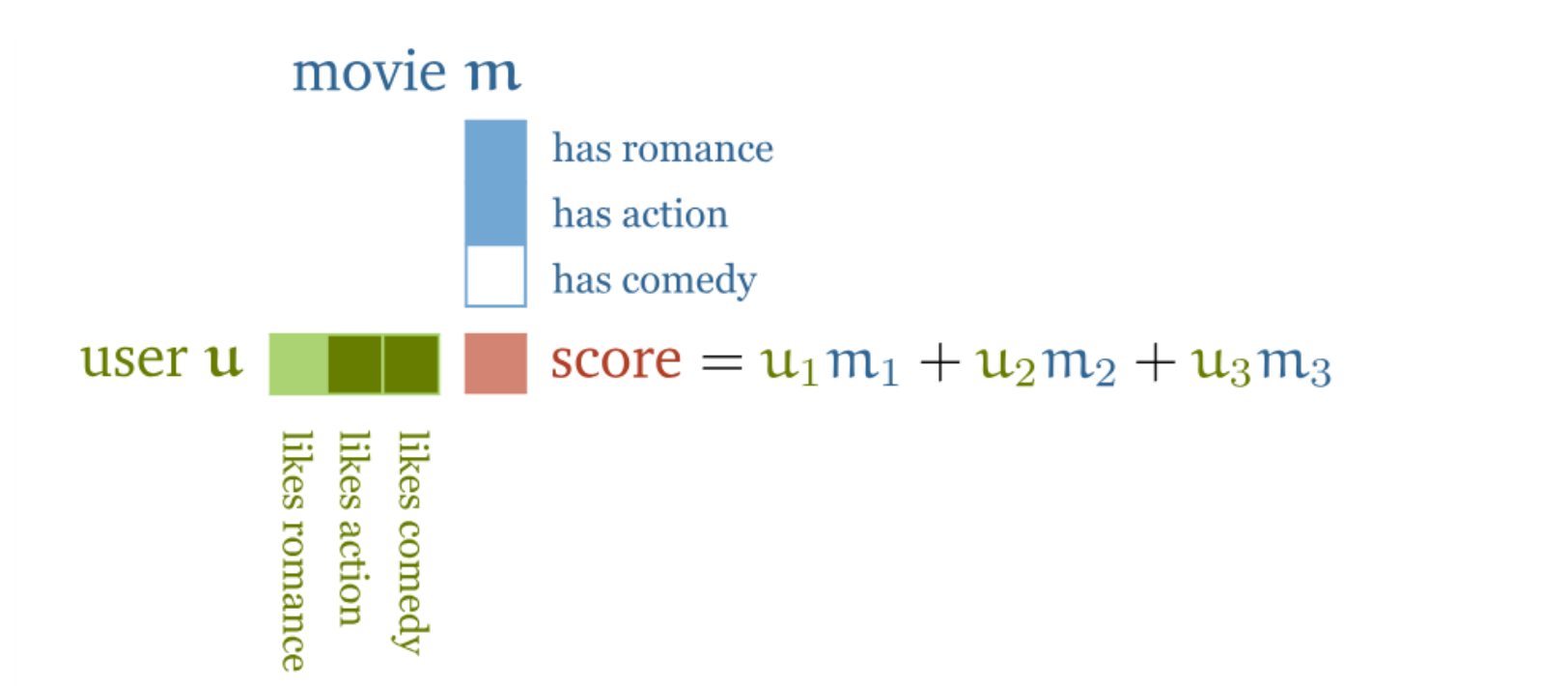
encoder



attention
decoder

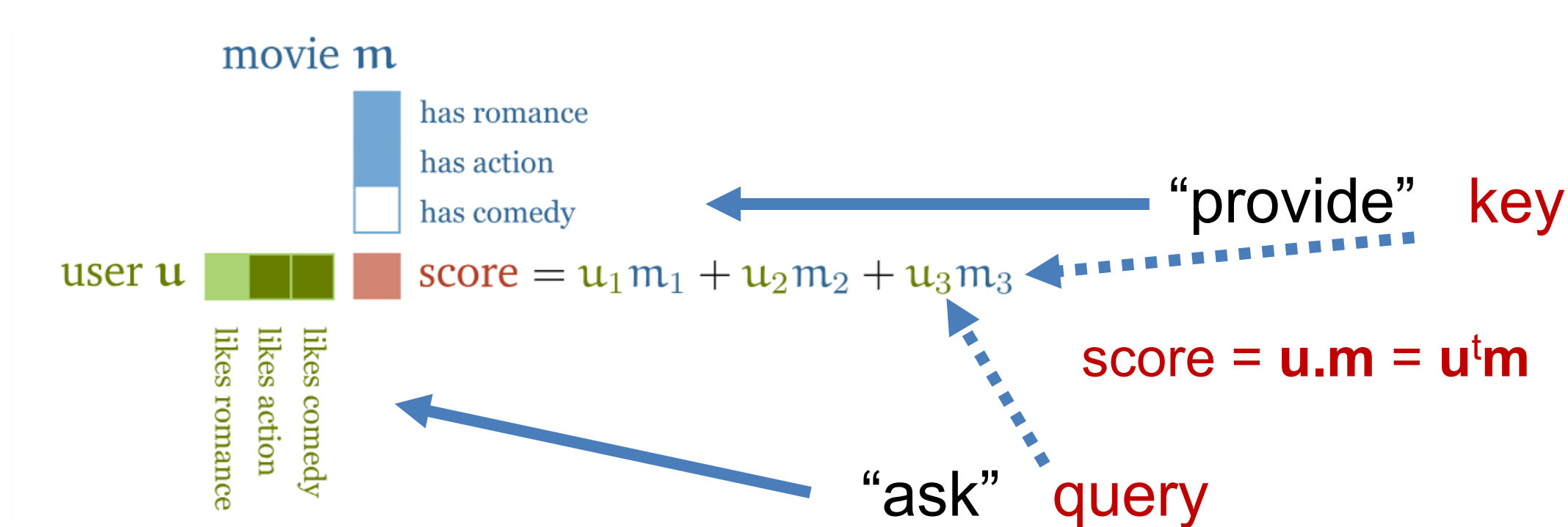
Understanding why self-attention works, in particular the dotprot construction

See <https://peterbloem.nl/blog/transformers>



Understanding why self-attention works, in particular the dotprot construction

See <https://peterbloem.nl/blog/transformers>

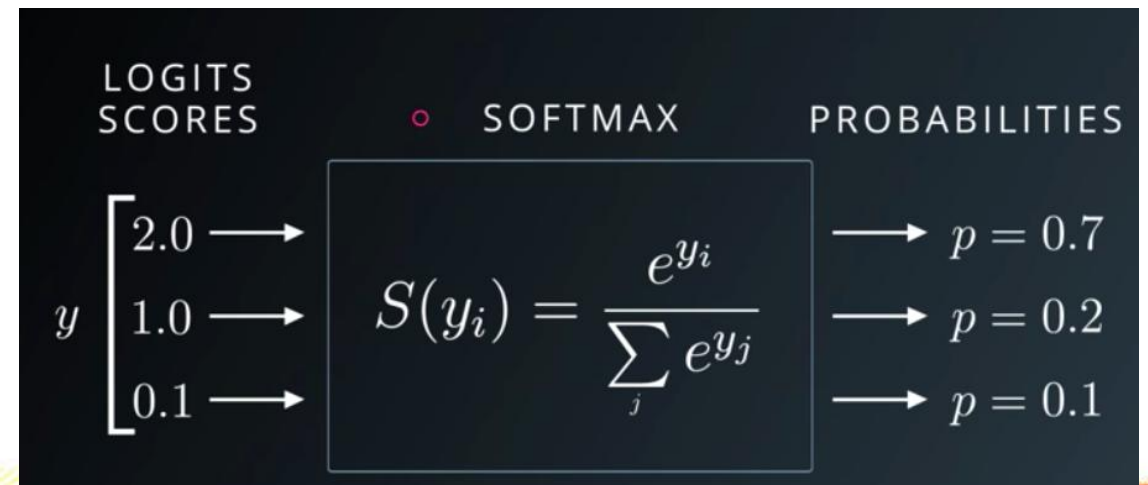


ATTENTION

- the dot product gives values anywhere between negative and positive infinity
- next, apply a **softmax**
 - to map all these values to $[0,1]$ and
 - to ensure that they sum to 1 over the whole sequence

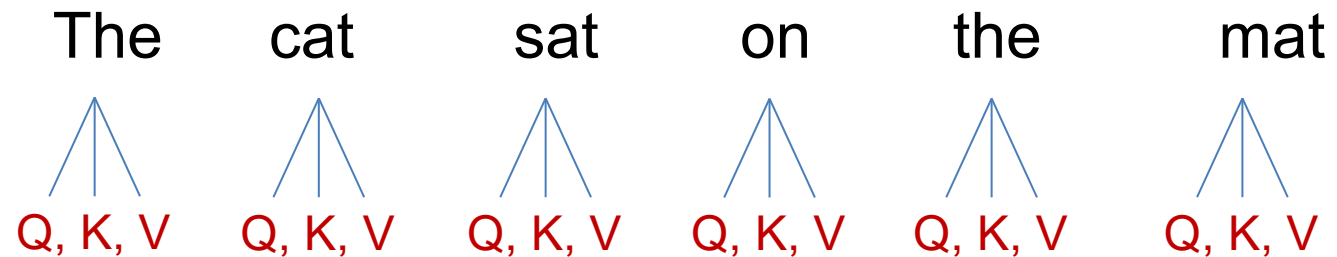
$$a_i = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}$$

This sequence is the basic operation of self attention.



Q, K, V

- **Q**uery: the degree a word looks for other words to pay attention to
- **K**ey: the degree to which a word likes to be paid attention to
- **V**alue: the value of the information of that word



A survey of transformers

Tianyang Lin, Yuxin Wang, Xiangyang Liu, Xipeng Qiu *

School of Computer Science, Fudan University, Shanghai, 200433, China

Shanghai Key Laboratory of Intelligent Information Processing, Shanghai, 200433, China

2.1.1. Attention modules

Transformer adopts attention mechanism with Query–Key–Value (QKV) model. Given the packed matrix representations of queries $\mathbf{Q} \in \mathbb{R}^{N \times D_k}$, keys $\mathbf{K} \in \mathbb{R}^{M \times D_k}$, and values $\mathbf{V} \in \mathbb{R}^{M \times D_v}$, the scaled dot-product attention used by Transformer is given by¹

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_k}}\right)\mathbf{V} = \mathbf{A}\mathbf{V}, \quad (1)$$

¹ If not stated otherwise, we use row-major notations throughout this survey (e.g., the i th row in \mathbf{Q} is the query \mathbf{q}_i) and all the vectors are row vectors by default.

$$\text{MultiHeadAttn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \dots, \text{head}_H)\mathbf{W}^O, \quad (2)$$

$$\text{where head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V). \quad (3)$$

In Transformer, there are three types of attention in terms of the source of queries and key–value pairs:

- *Self-attention*. In Transformer encoder, we set $\mathbf{Q} = \mathbf{K} = \mathbf{V} = \mathbf{X}$ in Eq. (2), where \mathbf{X} is the outputs of the previous layer.
- *Masked Self-attention*. In the Transformer decoder, the self-attention is restricted such that queries at each position can only attend to all key–value pairs up to and including that position. To enable parallel training, this is typically done by applying a mask function to the unnormalized attention matrix $\hat{\mathbf{A}} = \exp(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_k}})$, where the illegal positions are masked out by setting $\hat{A}_{ij} = -\infty$ if $i < j$. This kind of self-attention is often referred to as autoregressive or causal attention.²

FROM A MATHEMATICAL POINT OF VIEW

johnthickstun.com/doc
s/transformers.pdf

The Transformer Model in Equations

John Thickstun

Abstract

This document presents a precise mathematical definition of the transformer model introduced by Vaswani et al. [2017], along with some discussion of the terminology and intuitions commonly associated with the transformer. We also draw some connections between the transformer and lstm, based on observations by Levy et al. [2018].

1 Introduction

A **transformer block** is a parameterized function class $f_\theta : \mathbb{R}^{n \times d} \rightarrow \mathbb{R}^{n \times d}$. If $\mathbf{x} \in \mathbb{R}^{n \times d}$ then $f_\theta(\mathbf{x}) = \mathbf{z}$ where

$$Q^{(h)}(\mathbf{x}_i) = W_{h,q}^T \mathbf{x}_i, \quad K^{(h)}(\mathbf{x}_i) = W_{h,k}^T \mathbf{x}_i, \quad V^{(h)}(\mathbf{x}_i) = W_{h,v}^T \mathbf{x}_i, \quad W_{h,q}, W_{h,k}, W_{h,v} \in \mathbb{R}^{d \times k}, \quad (1)$$

$$\alpha_{i,j}^{(h)} = \text{softmax}_j \left(\frac{\langle Q^{(h)}(\mathbf{x}_i), K^{(h)}(\mathbf{x}_j) \rangle}{\sqrt{k}} \right), \quad (2)$$

$$\mathbf{u}'_i = \sum_{h=1}^H W_{c,h}^T \sum_{j=1}^n \alpha_{i,j}^{(h)} V^{(h)}(\mathbf{x}_j), \quad W_{c,h} \in \mathbb{R}^{k \times d}, \quad (3)$$

$$\mathbf{u}_i = \text{LayerNorm}(\mathbf{x}_i + \mathbf{u}'_i; \gamma_1, \beta_1), \quad \gamma_1, \beta_1 \in \mathbb{R}^d, \quad (4)$$

$$\mathbf{z}'_i = W_2^T \text{ReLU}(W_1^T \mathbf{u}_i), \quad W_1 \in \mathbb{R}^{d \times m}, W_2 \in \mathbb{R}^{m \times d}, \quad (5)$$

$$\mathbf{z}_i = \text{LayerNorm}(\mathbf{u}_i + \mathbf{z}'_i; \gamma_2, \beta_2), \quad \gamma_2, \beta_2 \in \mathbb{R}^d. \quad (6)$$

The notation softmax_j indicates we take the softmax (defined in Equation 9) over the d -dimensional vector indexed by j . The LayerNorm function [Lei Ba et al., 2016] is defined for $\mathbf{z} \in \mathbb{R}^k$ by

$$\text{LayerNorm}(\mathbf{z}; \gamma, \beta) = \gamma \frac{(\mathbf{z} - \mu_{\mathbf{z}})}{\sigma_{\mathbf{z}}} + \beta, \quad \gamma, \beta \in \mathbb{R}^k. \quad (7)$$

$$\mu_{\mathbf{z}} = \frac{1}{k} \sum_{i=1}^k \mathbf{z}_i, \quad \sigma_{\mathbf{z}} = \sqrt{\frac{1}{k} \sum_{i=1}^k (\mathbf{z}_i - \mu_{\mathbf{z}})^2}. \quad (8)$$

Many pretrained models available on HuggingFace

HuggingFace: repository with available models (from OpenAI, Meta, ..., and other groups)

- April 9, 2025: 1,582,000 models (nearly 50 subcategories); 352000 datasets (8 different modalities)
- Sept 30, 2025: >2,000,000 models

Examples:

- A Wav2Vec2 model that is pretrained on relatively clean LibriSpeech data
- A Wav2Vec2 model that is pretrained and fine-tuned on noisy + clean data
- XLS-R: multilingual and multi-accent, pretrained on the CommonVoice corpus
- A huBERT model, improves or matches Wav2Vec2 performance and extends the Wav2Vec2 architecture, pretrained on LibriSpeech data
- Whisper, Whisper-large-v3, etc. etc.

Figures taken from various sources, including

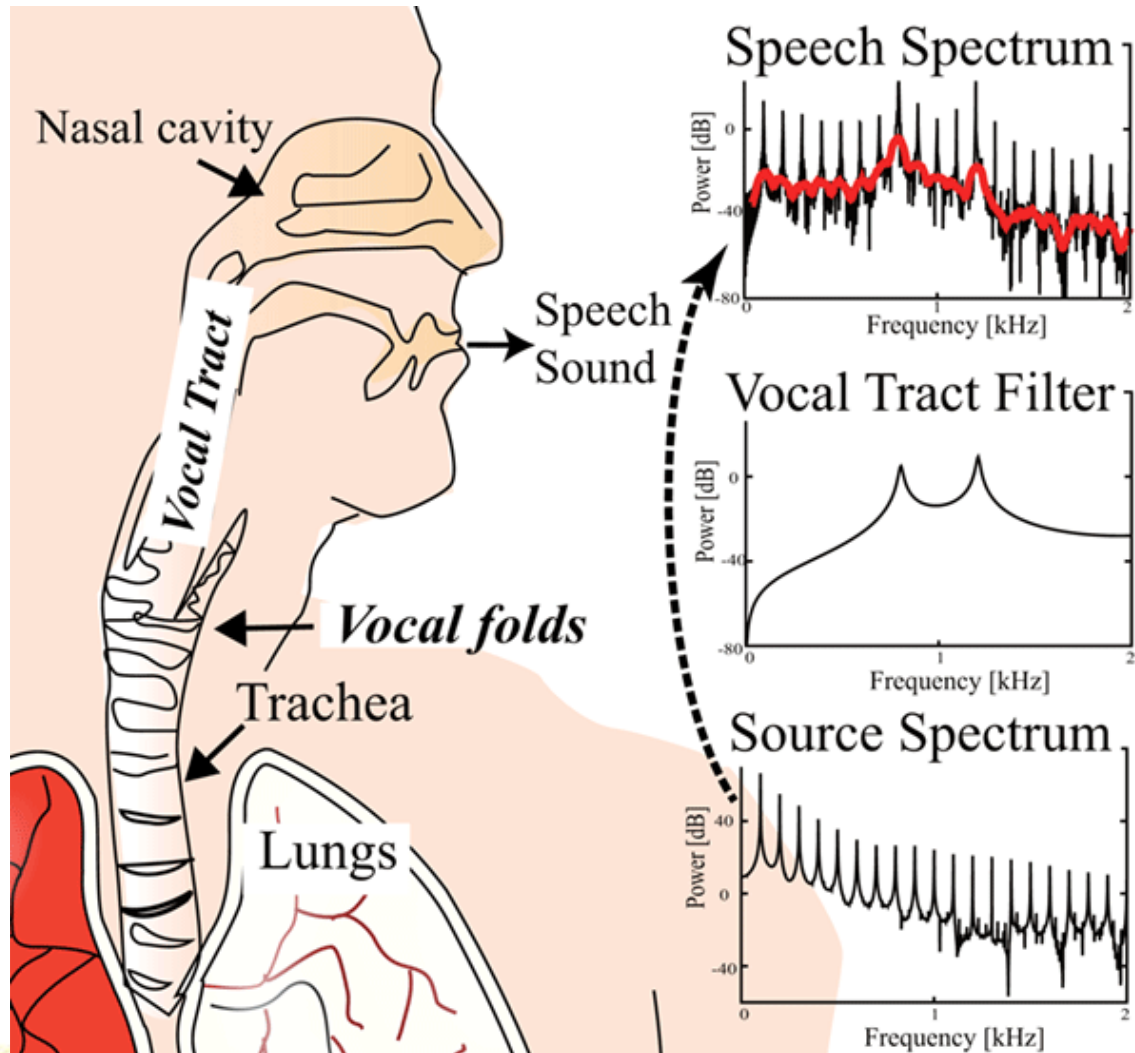
- Jakob Poncelet, Doctoral Dissertation, Leuven, Belgium, February 2025.
- J. Kim, J. Hong, and H. Park. “Prospects of deep learning for medical imaging”. In: *Precision and Future Medicine* 2.2 (2018), pp. 37–52. doi: 10.23838/pfm. 2018.00030.
- H. Yonekawa, S. Sato, and H. Nakahara. “A Ternary Weight Binary Input Convolutional Neural Network: Realization on the Embedded Processor”. In: *Proc. IEEE Int. Symp. on Multiple-Valued Logic (ISMVL)*. 2018, pp. 174–179. doi: 10.1109/ISMVL.2018.00038.
- V. Peddinti, D. Povey, and S. Khudanpur. “A time delay neural network architecture for efficient modeling of long temporal contexts”. In: *Proc. Interspeech*. 2015, pp. 3214–3218. doi: 10.21437/Interspeech.2015-647.
- J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies*. 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423.
- S. Chen et al. “WavLM: Large-scale self-supervised pre-training for full stack speech processing”. In: *IEEE Journal of Selected Topics in Signal Processing (JSTSP)* 16.6 (2022), pp. 1505–1518. doi: 10.1109/JSTSP.2022.3188113.

ASR:

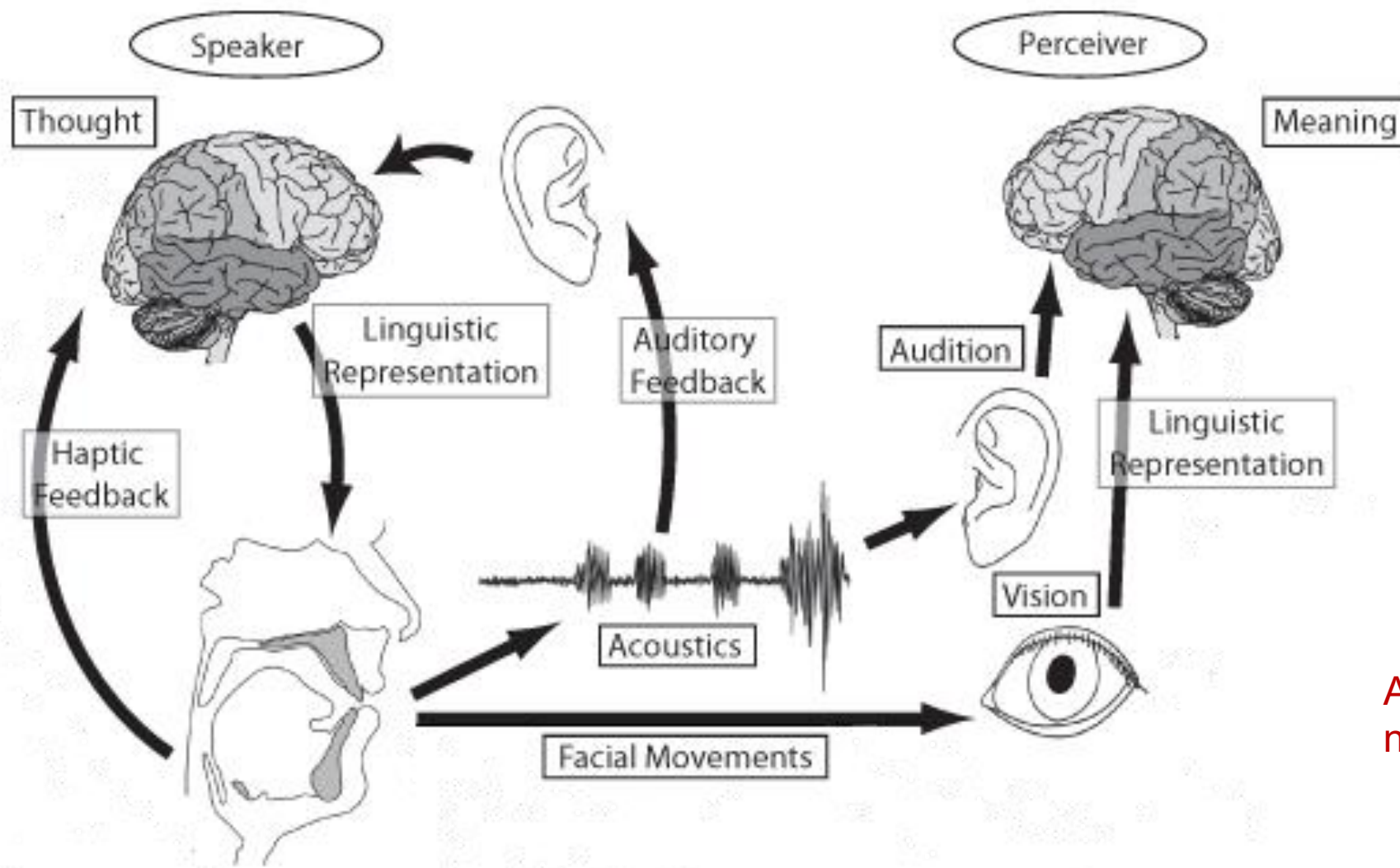
It all starts at the

source...

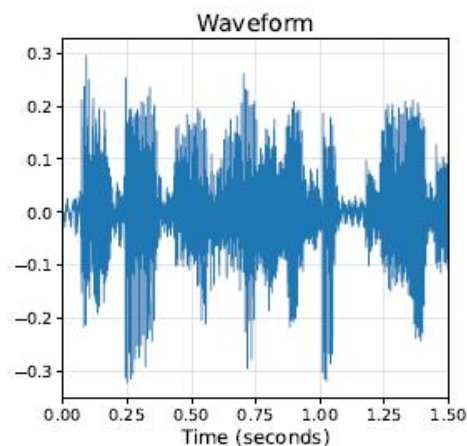




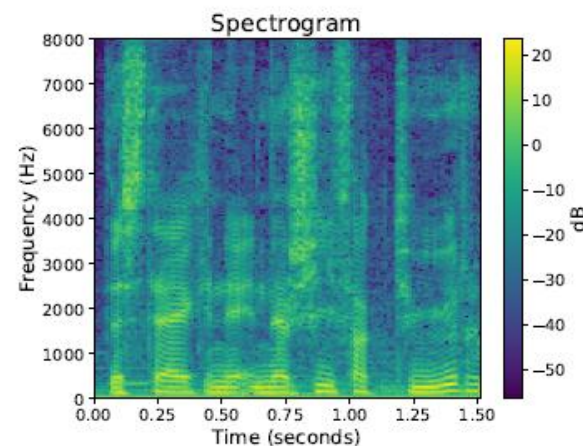
Borden & Harris (textbook)



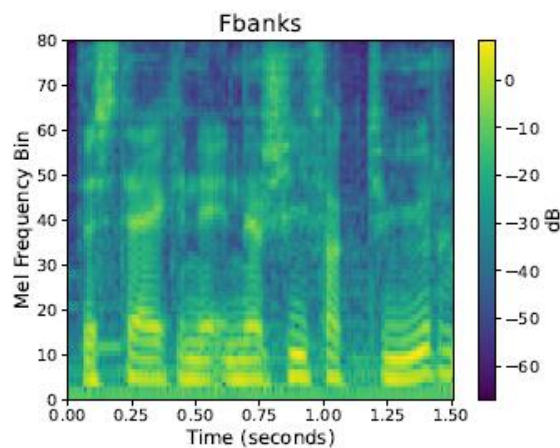
Almost all arrows involve non-linear processes



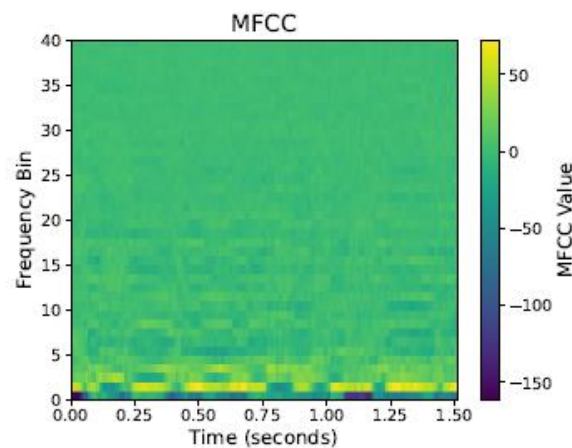
(a) raw waveform signal



(b) 256-dimensional spectrogram



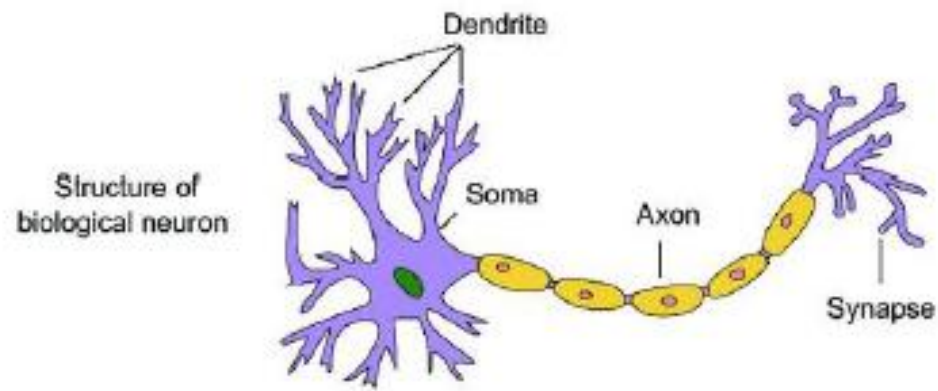
(c) 80-dimensional Fbanks



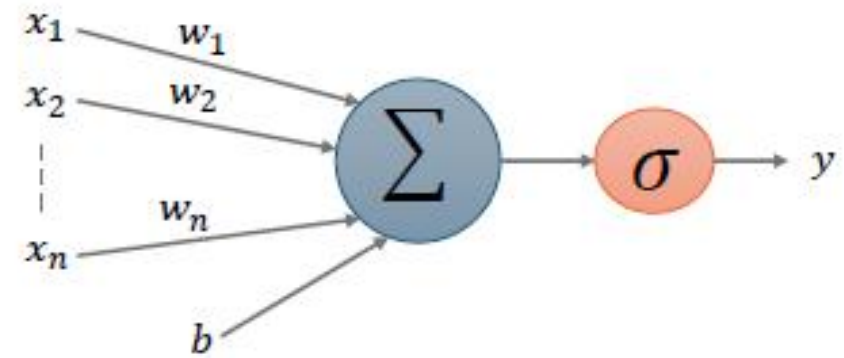
(d) 40-dimensional MFCCs

Optimal feature representation of audio depends on the task

Bengio et al., 2013:
representation learning



(a) Brain neuron



(b) Perceptron

In a general neural network, such a building block is called a **neuron**. Like a perceptron, a neuron receives inputs, has a set of weights and a bias, and computes an activation (which is generally not constrained to 0 or 1). Mathematically, the output y of a **neuron** can be expressed in terms of its inputs x_i , weights w_i , bias term b and activation function σ as in Eq. 2.1.

$$y = \sigma\left(\sum_{i=1}^n w_i x_i + b\right) \quad (2.1)$$

MLPs are general approximators when #hidden layers ≥ 2 (already known in the late 80-ies)
Weakness: context

Multi-layer Perceptron

Typically, a neural network consists of multiple layers of neurons stacked on top of each other, which in its simplest form is referred to as **multi-layer perceptron (MLP)**. An MLP consists of several layers of neurons, where every neuron in a layer is connected to neurons in the layer before (i.e. a *fully connected* layer). Such a network is also called a **Feed-forward Neural Network**. The layers between the input layer and the output layer are called the **hidden layers**. Figure 2.6 shows a simple MLP with two hidden layers. In this example, there are four inputs and two outputs. In a speech recognition model (for which unfortunately for our environment, two hidden layers does not suffice), the input could for example be a time-frame of 80-dimensional fbanks (i.e. 80 input values), and the output could be for example 26 neurons, one for each letter of the alphabet.⁵

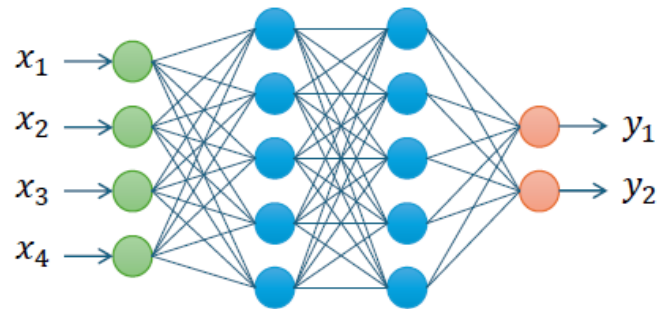
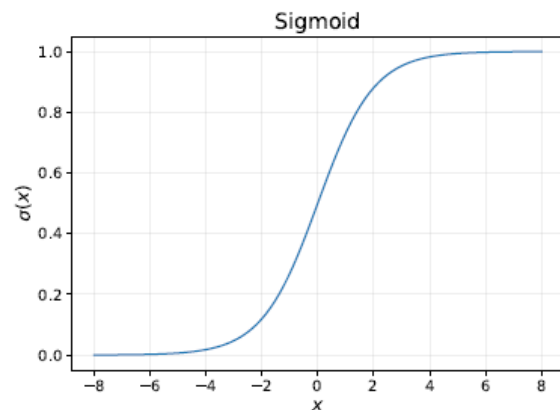
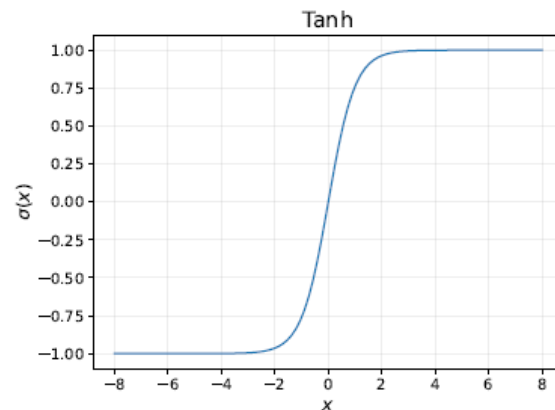


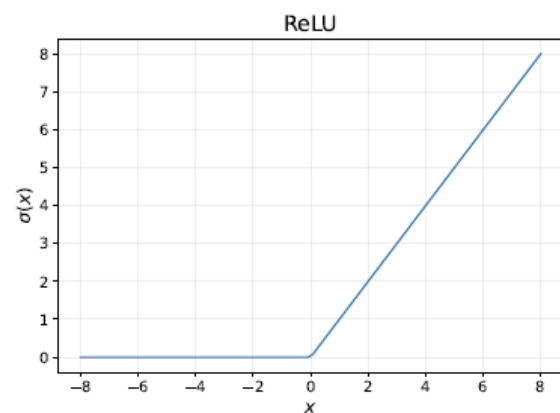
Figure 2.6: Multilayer Perceptron



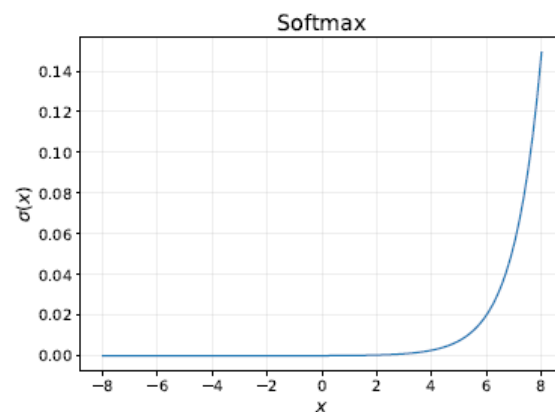
(a) $\sigma_{sigmoid}(x) = \frac{1}{1+e^{-x}}$



(b) $\sigma_{tanh}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$



(c) $\sigma_{ReLU}(x) = \max(0, x)$



(d) $\sigma_{softmax}(\mathbf{x})_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$

Figure 2.4: Common activation functions used in neural networks.

Classical:

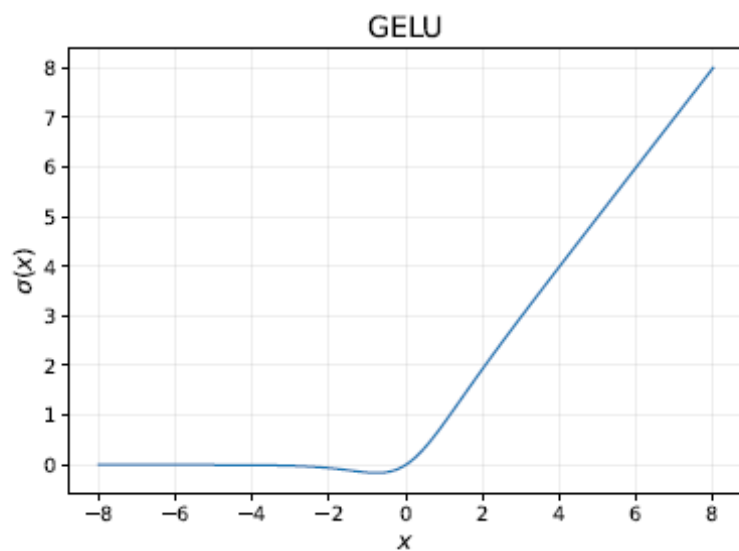
- sigmoid
- tanh

(often problems with vanishing gradient)

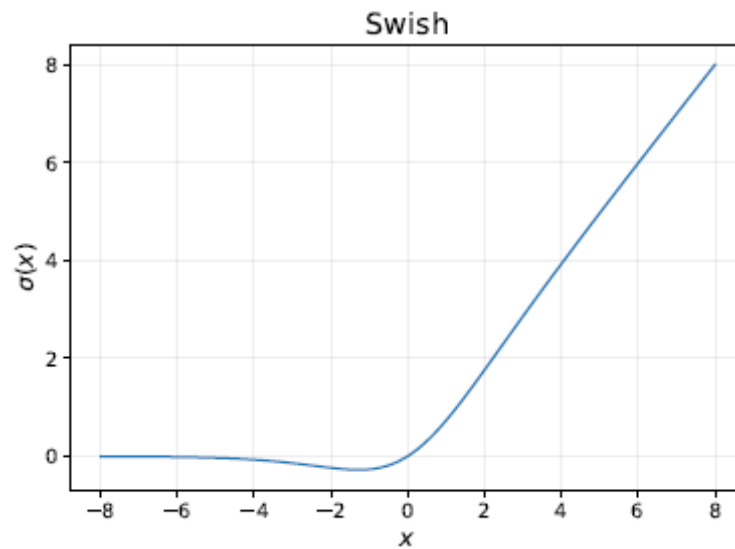
More recent:

- ReLU (rectified linear unit)
- GELU (Gaussian Error Linear Unit)
- ... many more

- Softmax in the last layer

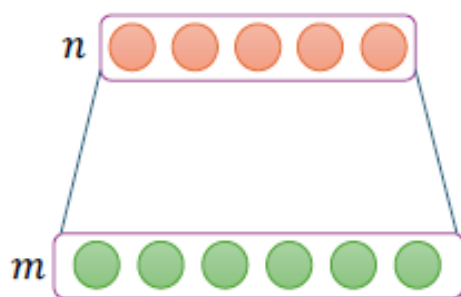


(a) $\sigma_{GELU}(x) = x\Phi(x)$

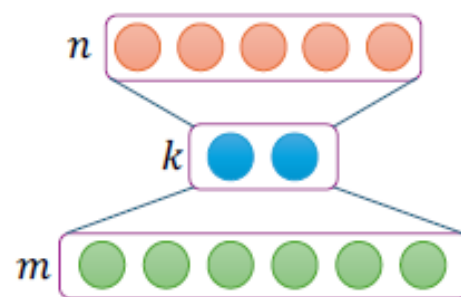


(b) $\sigma_{swish}(x) = \frac{x}{1+e^{-\beta x}}$

Figure 2.5: Common activation functions used in recent speech processing models.



(a) Standard layer



(b) Bottleneck layer

Figure 2.11: Example of a bottleneck layer. In the standard layer, there are $m \times n = 30$ connections. In the bottleneck layer, this is reduced to $(m \times k) + (k \times n) = 22$.

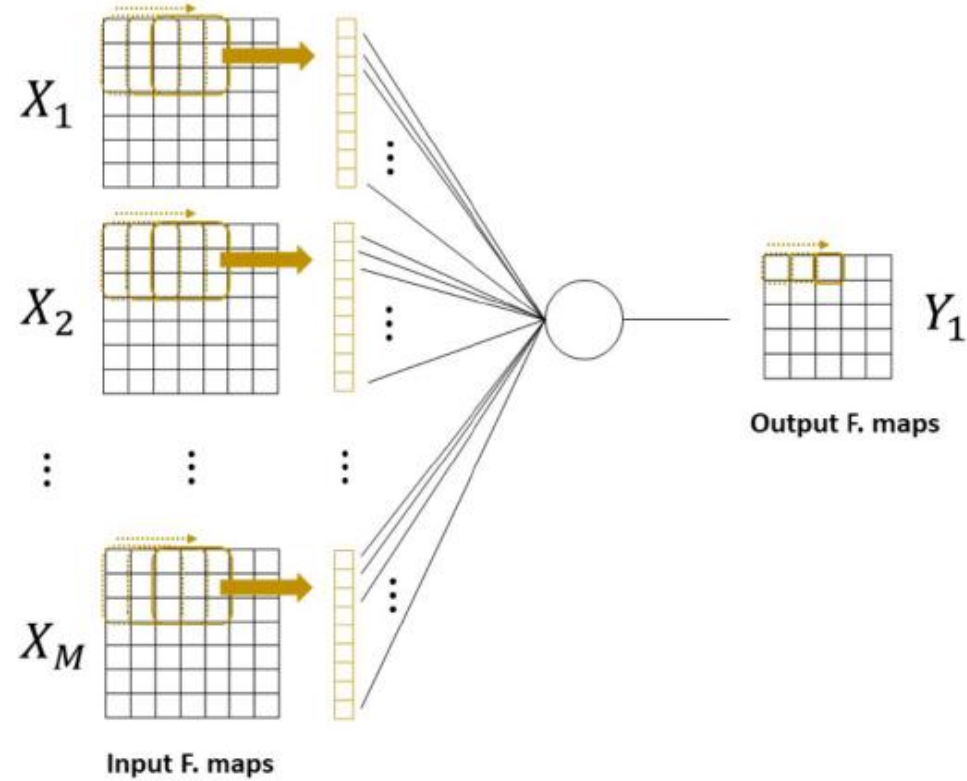
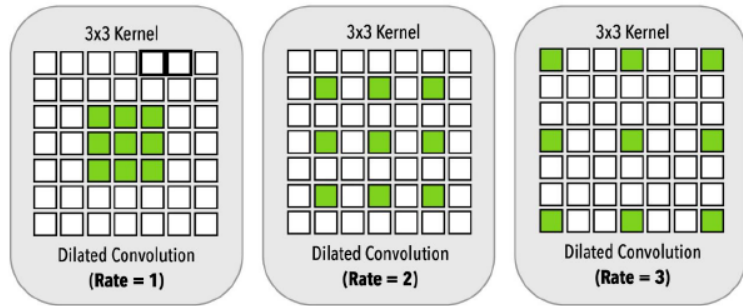
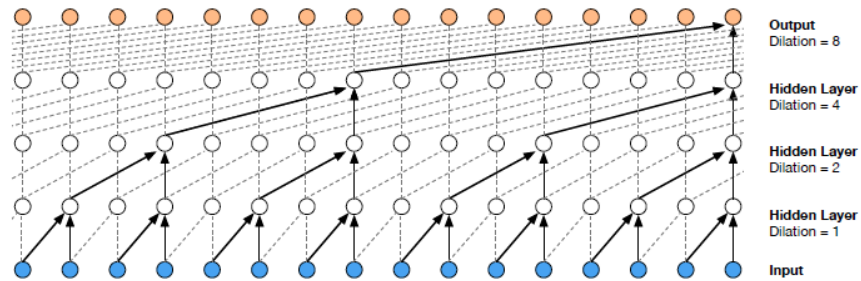


Figure 2.8: Illustration of a 2-dimensional convolutional layer. The filter is applied over the input by processing in a grid-like manner performing the multiplication and accumulation operation on local grids. Then the outputs of the filters are combined



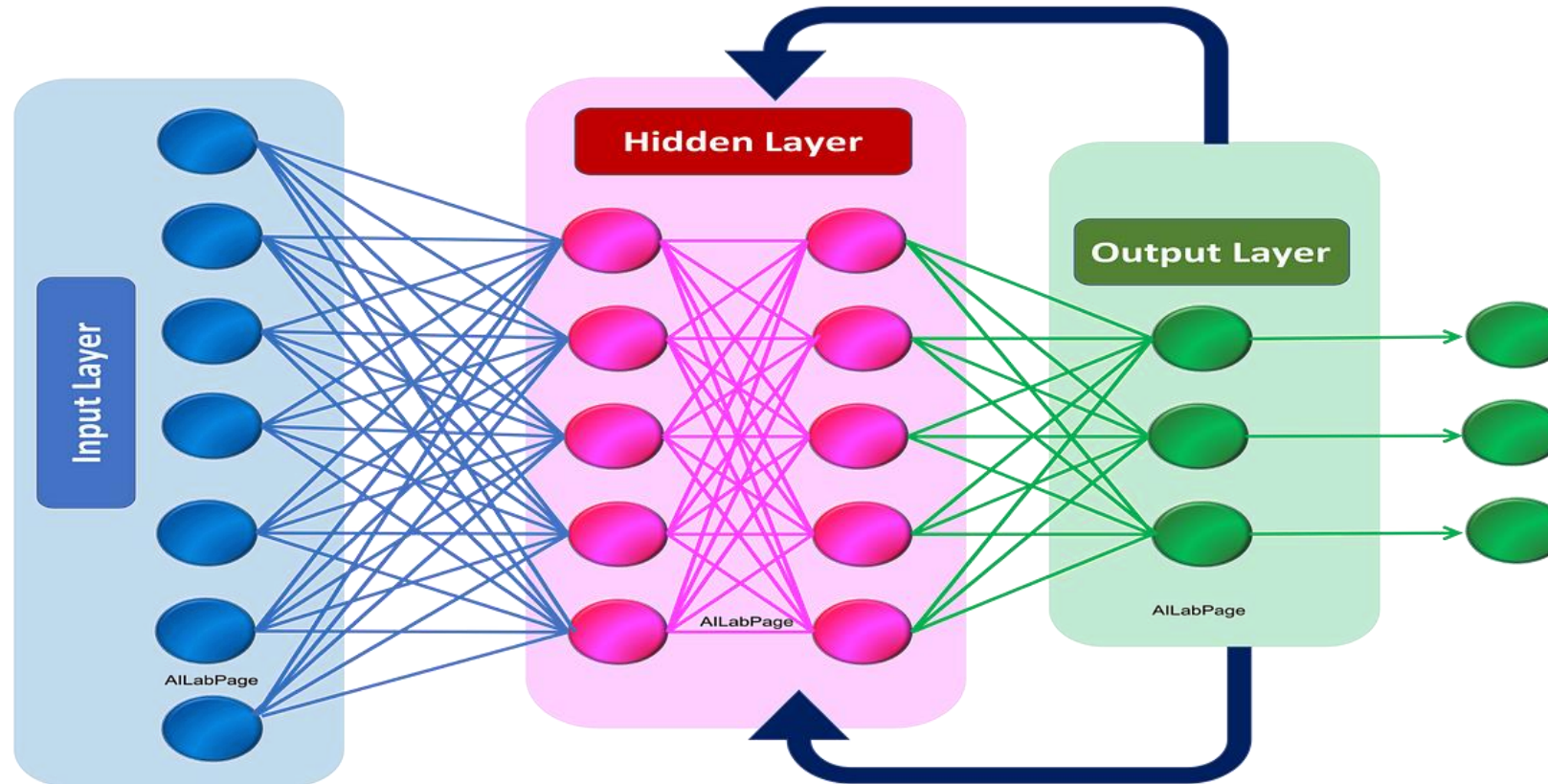
(a) 2D dilated convolution (from [102])



(b) 1D dilated convolution (from [101])

Figure 2.10: Illustration of dilated convolutions applied to (a) 2D convolutions and (b) 1D convolutions in WaveNet.

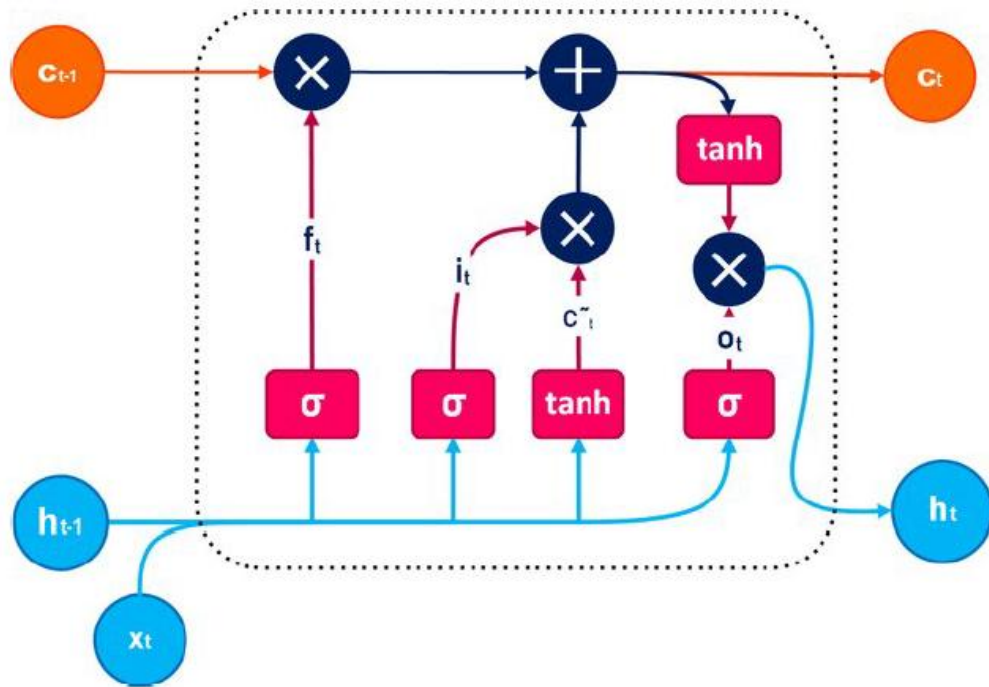
Recurrent Neural Networks



RNNs, LSTMs

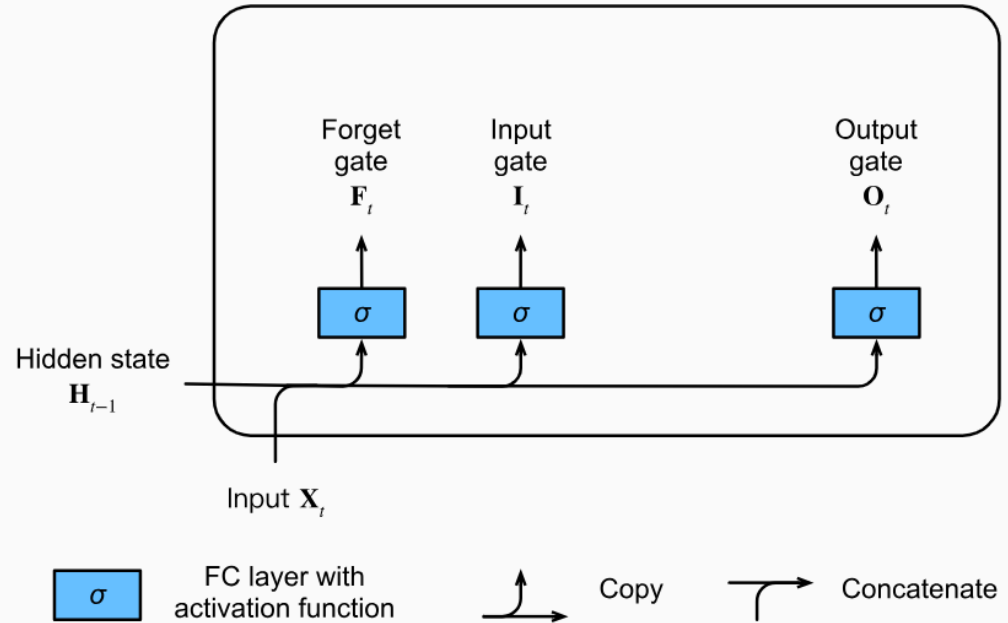
- RNNs: *vanishing gradients* and *exploding gradients*
 - “*vanishing gradient problem*”
- Improvement: **Long Short-Term Memory (LSTM)**
 - input gate, forget gate and output gate determine which information to store, discard and output respectively
 - LSTM can learn which information in the sequence might be required later on, and discard the rest, by separating the long-term and short-term memory
 - LSTMs may remember long-term dependencies

The key distinction between vanilla RNNs and LSTMs is that the latter support gating of the hidden state



Long Short-Term Memory (LSTM)

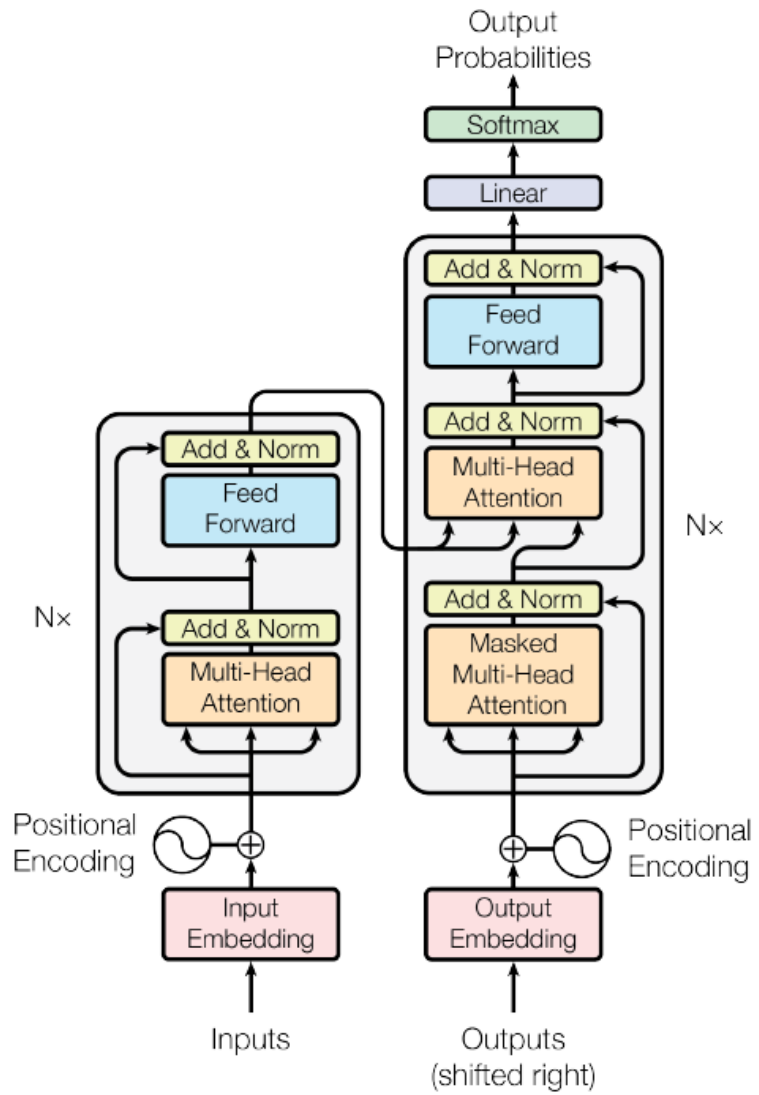
LSTMs are sometimes considered a member of the RNN family.
RNNs are still used for simpler sequential tasks in smaller, resource-constrained environments.



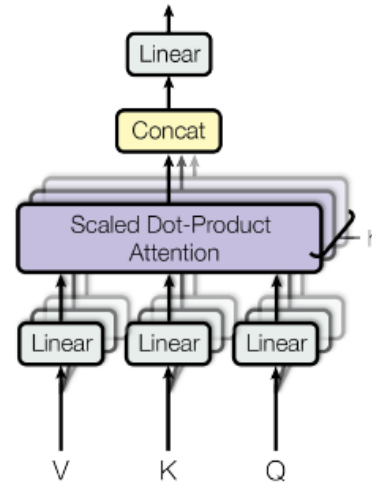
Computing the input gate, the forget gate, and the output gate in an LSTM model.

Transformers (2015-6-7)

- led to enormous breakthroughs in deep learning
- natural language processing, computer vision and speech processing
- uses parallelisation and attention mechanisms to improve upon RNNs for sequence modelling tasks
- encoder-decoder architecture



(a) Transformer Network



(b) Multi-head Attention

End-to-end (E2E)

- The first approach for E2E ASR is based on **Connectionist Temporal Classification (CTC)**
- CTC directly maps an input sequence to an output sequence, without requiring an alignment between the two.
- CTC predicts the output label probabilities for every input timestep, including the possibility of repetitions and a *blank* token (often ‘_’) (**logit matrix**)
- CTC assumes that the relation between input and output is *monotonic*
 - i.e. if a word comes after another word in the output sequence, it must also come after this word in the input sequence

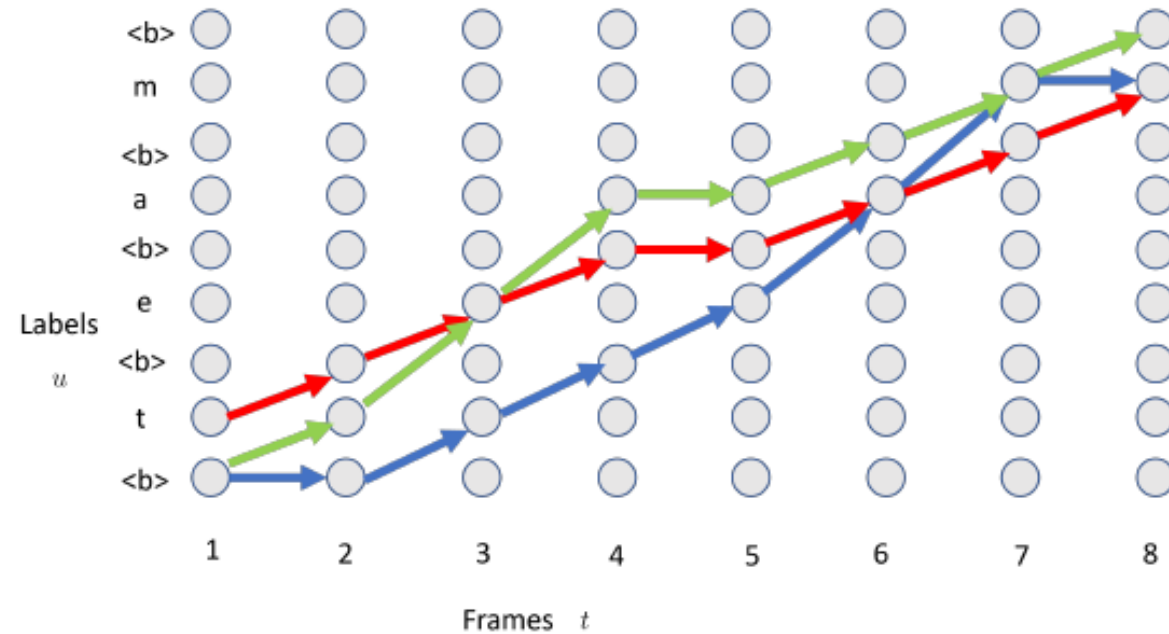


Figure 2.22: Example of CTC alignments. The figure shows three possible valid paths which make up the word “team”. The acoustic frame indices are plotted on the horizontal axis and the output letters on the vertical axis. The $\langle b \rangle$ represents the blank label ‘_’. The red path is $(t_e_a_m)$, the green path is $(_teaa_m_)$ and the blue path is $(_t_eammm)$. Figure from [161].

Connectionist Temporal Classification (CTC)

- **CTC computes the alignment between the input speech signal and the output sequence of the words**
- CTC uses a blank label – that represents a pause or a transition between words or phonemes.
- For a given transcription sequence, there are several possible alignments
 - (a, -, b, c, -, -) (a, -, b, c, -, -) and (-, a, -, b, -, c, -, -) (a, b, c)
 - both correspond to the character sequence (a, b, c) (a, b, c)
- CTC kicks out doubles
- **CTC aims to maximize the total probability of the c to get the correct output word sequence**
 - it doesn't require prior segmentation or alignment of the data



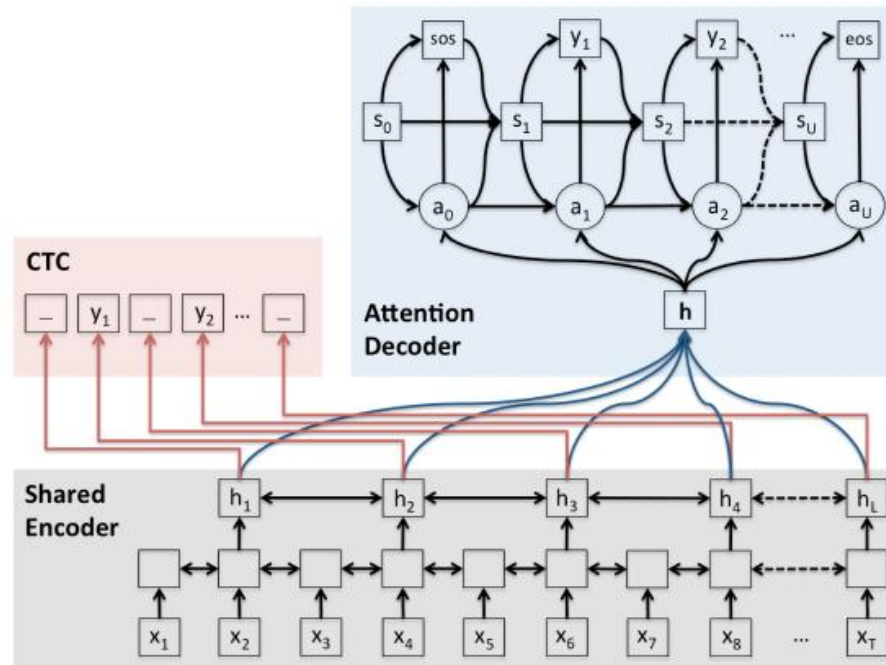


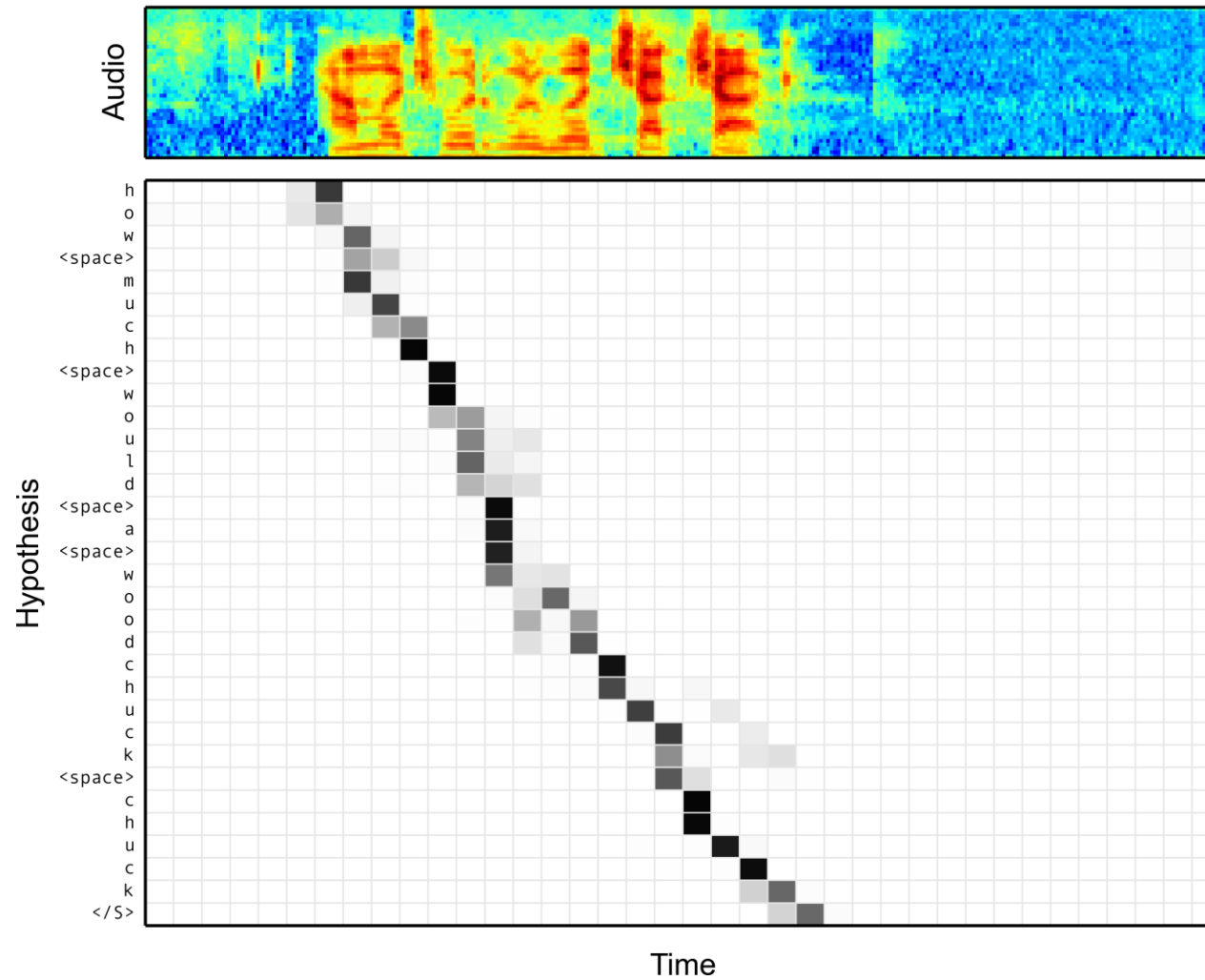
Figure 2.23: Schematic example of an RNN-based hybrid CTC/Attention model (CTC+AED) speech recognition model. Inputs and outputs are denoted x_i and y_i respectively, the decoder cell states are denoted s_i . a_i represents the context vector for step i which is computed from the encoder states h_t using the attention weights $\alpha_{i,t}$. Figure from [167].

Many combinations are possible.

Here an RNN in combination with CTC

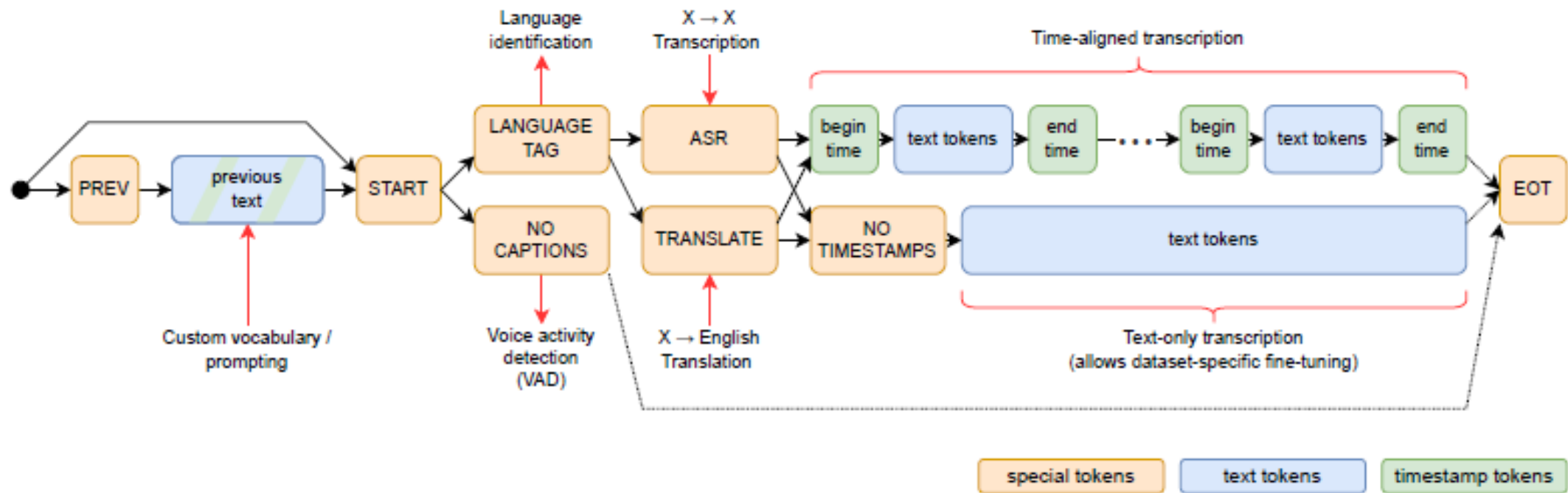
(see Jakob Poncelet's thesis for more details)

Alignment between the Characters and Audio



Whisper (2022) + several variants

- major impact
- speech recognition and speech translation model (680 thousand hours, 5 million hours of speech)
- training based on ***weak supervision*** (i.e., the transcripts not perfect)
- **robust** against various domain factors -- too robust?
- **decoder** is extended to many down-stream tasks, including language identification, timing generation, prompting, speech translation (to English), speech recognition.
- *task-specific tokens*: if the decoder has to transcribe, then the first token in the decoder is “|transcribe|”. If it has to translate, the first token is “|translate|”
- the decoder is conditioned on the previous sentence as a *prompt*



Thank you for your attention