Structure discovery by Transformer networks

Very Short history … deep learning

- MultiLayer Perceptron: since 1958, Rosenblatt. Since 2009: theory behind training deep models (Bengio, et al.)
- RNN (recurrent neural networks): (1925, Ising), 1980, various people
- *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)
- 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)

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

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

ANOTHER KEY PAPER (2022)

AI Open 3 (2022) 111–132

Contents lists available at ScienceDirect

AI Open

journal homepage: www.keaipublishing.com/en/journals/ai-open

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

ARTICLE INFO

Keywords:
Transformer
Self-attention
Pre-trained models
Deep learning

ABSTRACT

Transformers have achieved great success in many artificial intelligence fields, such as natural language processing, computer vision, and audio processing. Therefore, it is natural to attract lots of interest from academic and industry researchers. Up to the present, a great variety of Transformer variants (a.k.a. X-formers) have been proposed, however, a systematic and comprehensive literature review on these Transformer variants is still missing. In this survey, we provide a comprehensive review of various X-formers. We first briefly introduce the vanilla Transformer and then propose a new taxonomy of X-formers. Next, we introduce the various X-formers from three perspectives: architectural modification, pre-training, and applications. Finally, we outline some potential directions for future research.

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 $f_\theta : \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times d}$. If $\mathbf{x} \in \mathbb{R}^{n\times d}$ then $f_\theta(\mathbf{x}) = \mathbf{z}$ where

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

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

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

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

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

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

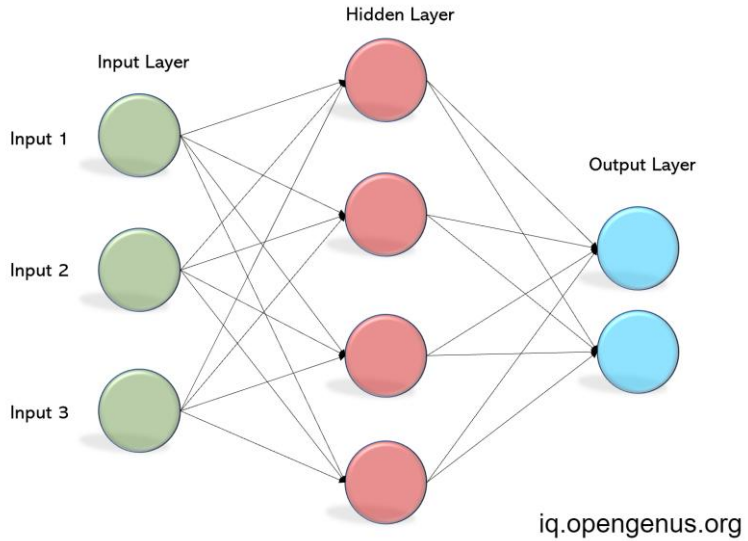
The notation softmax_j indicates we take the softmax (defined in Equation 9) over the d-dimensional vector indexed by j. The LayerNorm function [Lei Ba et al., 2016] is defined for $\mathbf{z} \in \mathbb{R}^k$ by

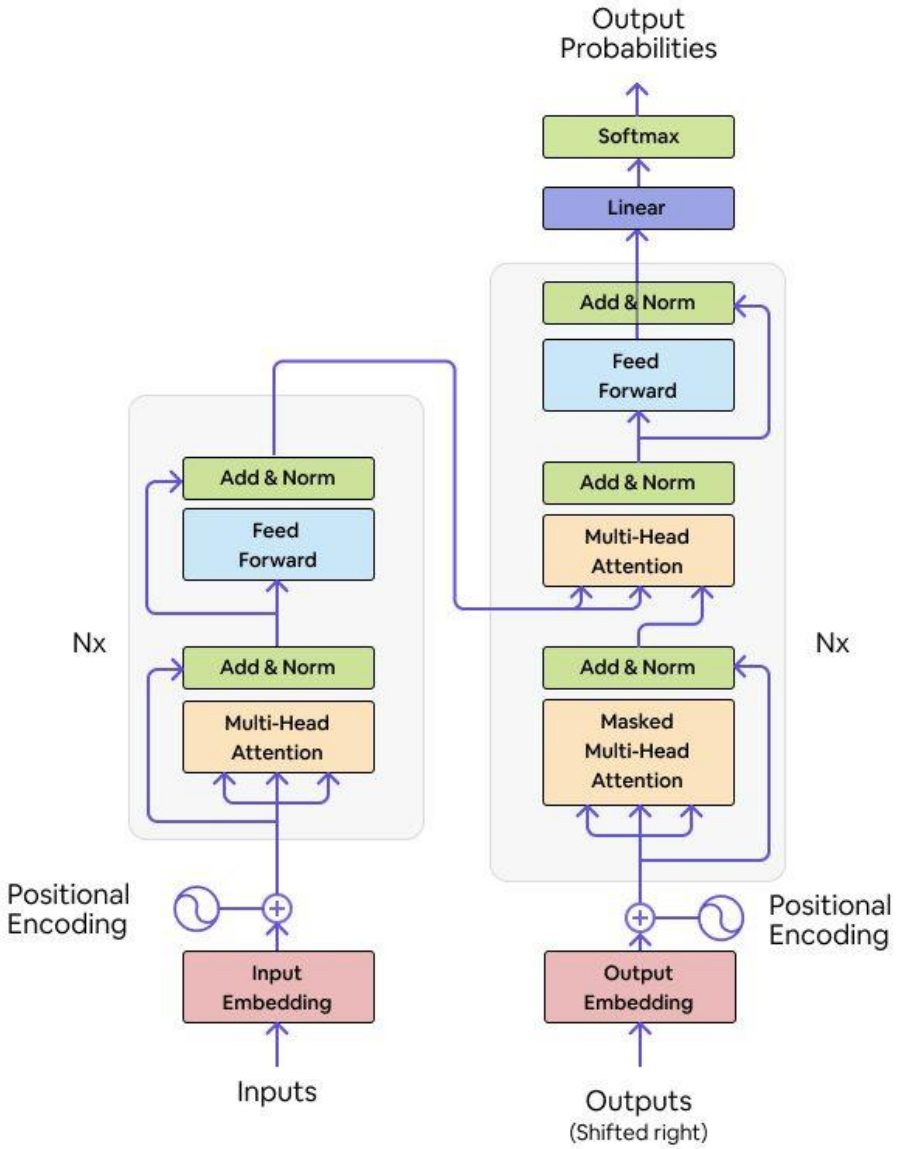
$$\text{LayerNorm}(\mathbf{z}; \gamma, \beta) = \gamma\frac{(\mathbf{z} - \mu_\mathbf{z})}{\sigma_\mathbf{z}} + \beta, \qquad \gamma, \beta \in \mathbb{R}^k. \tag{7}$$

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

The parameters θ consist of the entries of the weight matrices W, along with the LayerNorm parameters γ and β indicated on the right-hand side. The input $\mathbf{x} \in \mathbb{R}^{n\times d}$ should be interpreted as a collection of n objects, each with d features (often, but not always, a length-n sequence of d-vectors). Observe that the output $\mathbf{z} \in \mathbb{R}^{n\times d}$ has the same structure as the input $\mathbf{x} \in \mathbb{R}^{n\times d}$; a **transformer** is a composition of L transformer blocks, each with their own parameters: $f_{\theta_L} \circ \cdots \circ f_{\theta_1}(\mathbf{x}) \in \mathbb{R}^{n\times d}$.

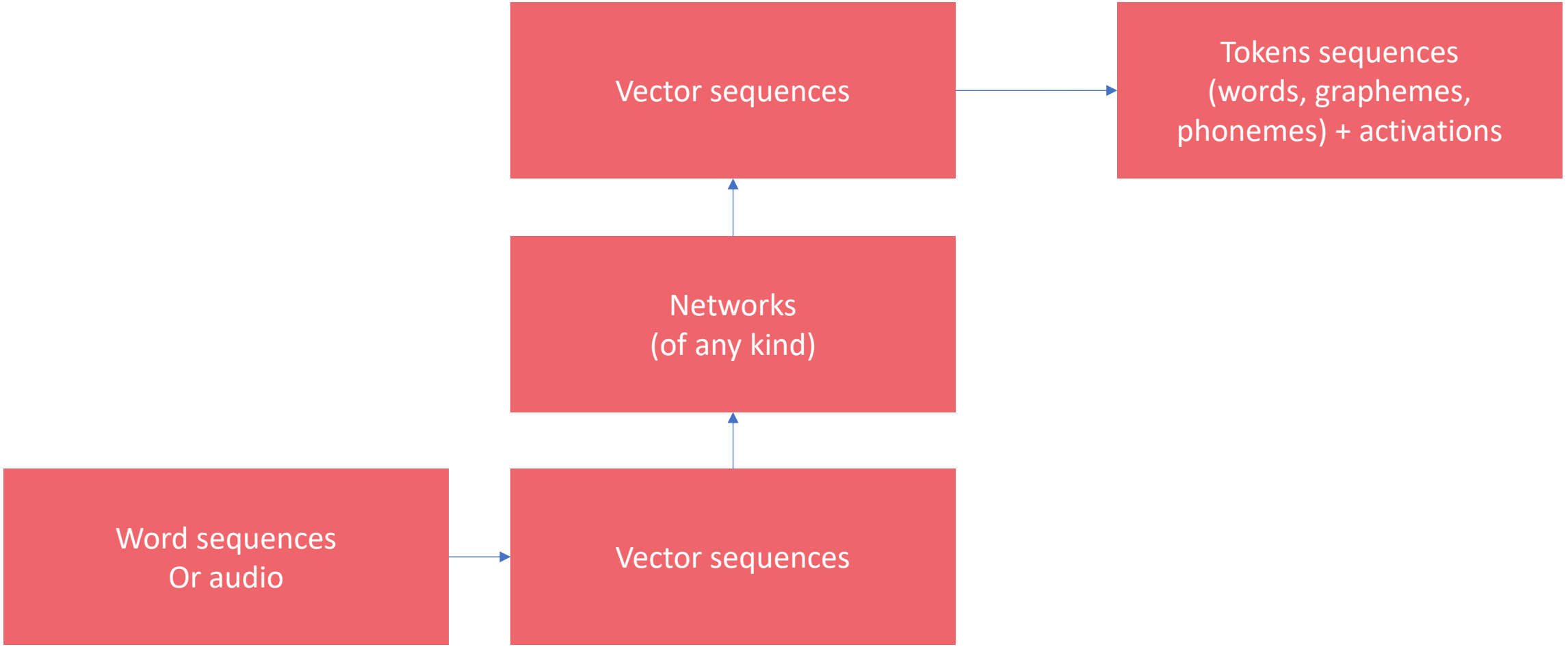
NETWORKS



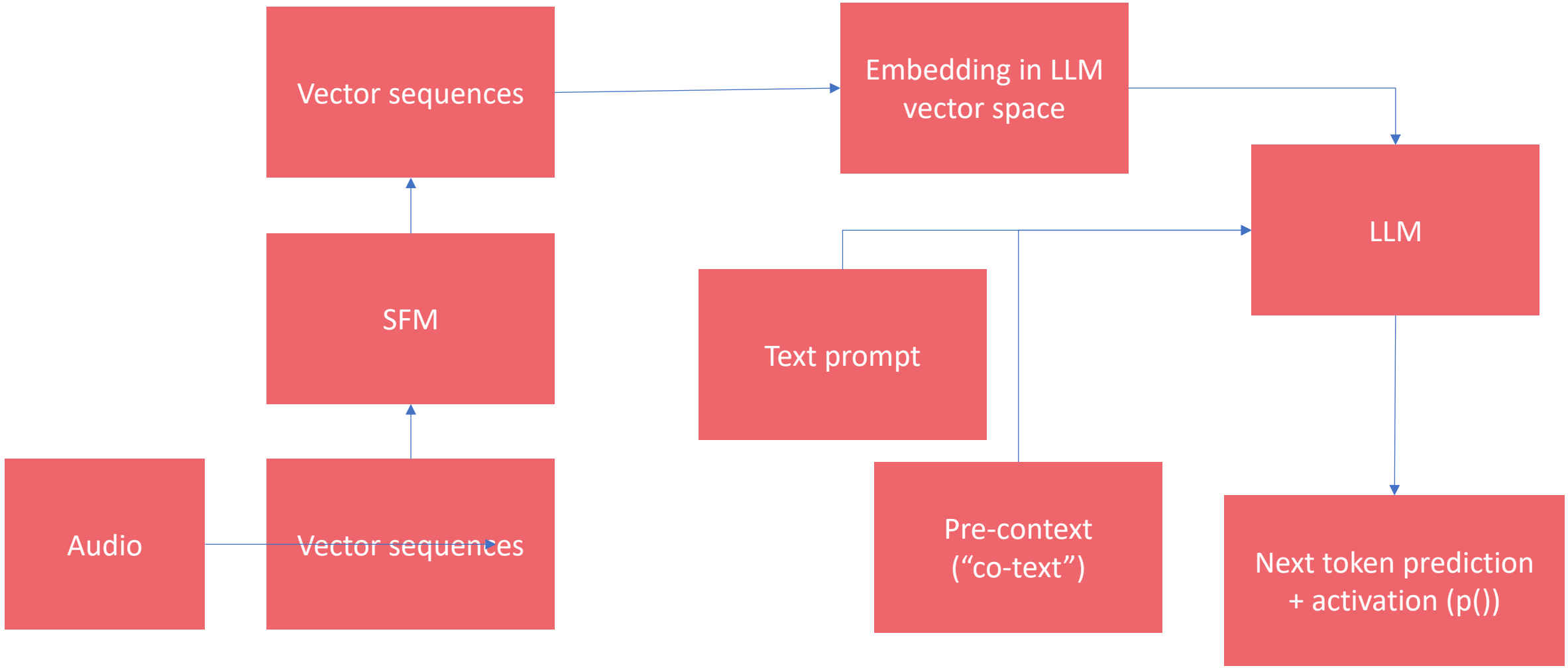


(From the Attention paper)

Vectors and networks



SFMs and LLMs – see e.g. Verdini et al. (2025)

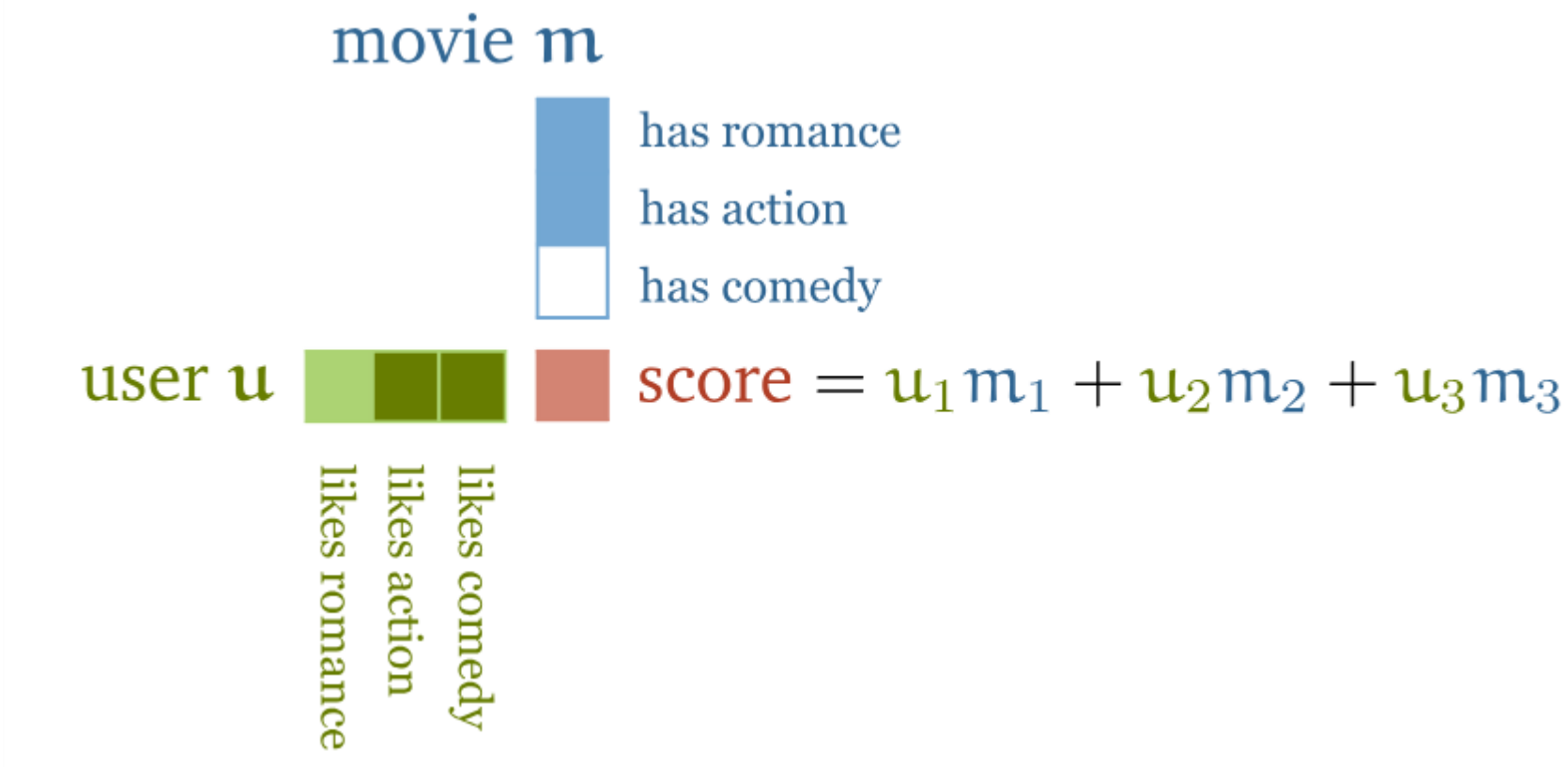


HOW DO WE GET THESE VECTORS?

- Large text -> word2vec -> vectors (one vector per word)
  - One per orthographic form, context-indep.
  - Context-dependent, e.g. homophones (*Eng.*) bank – bank
  - (not the scope of this course)
- Audio -> feature extraction -> vectors
  - “Spectral slices”
  - Typically one vector per 10 ms or 20 ms

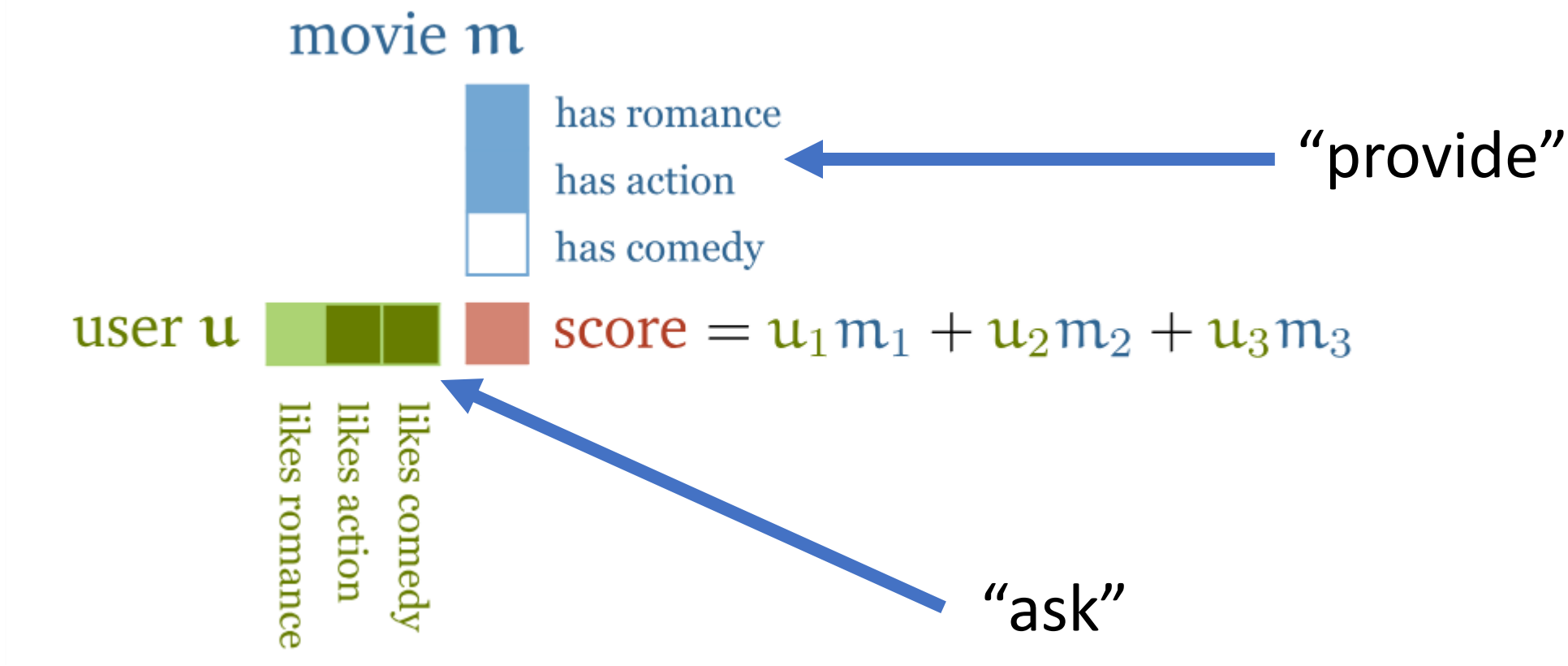
Understanding why self-attention works, in particular the dotprot construction

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



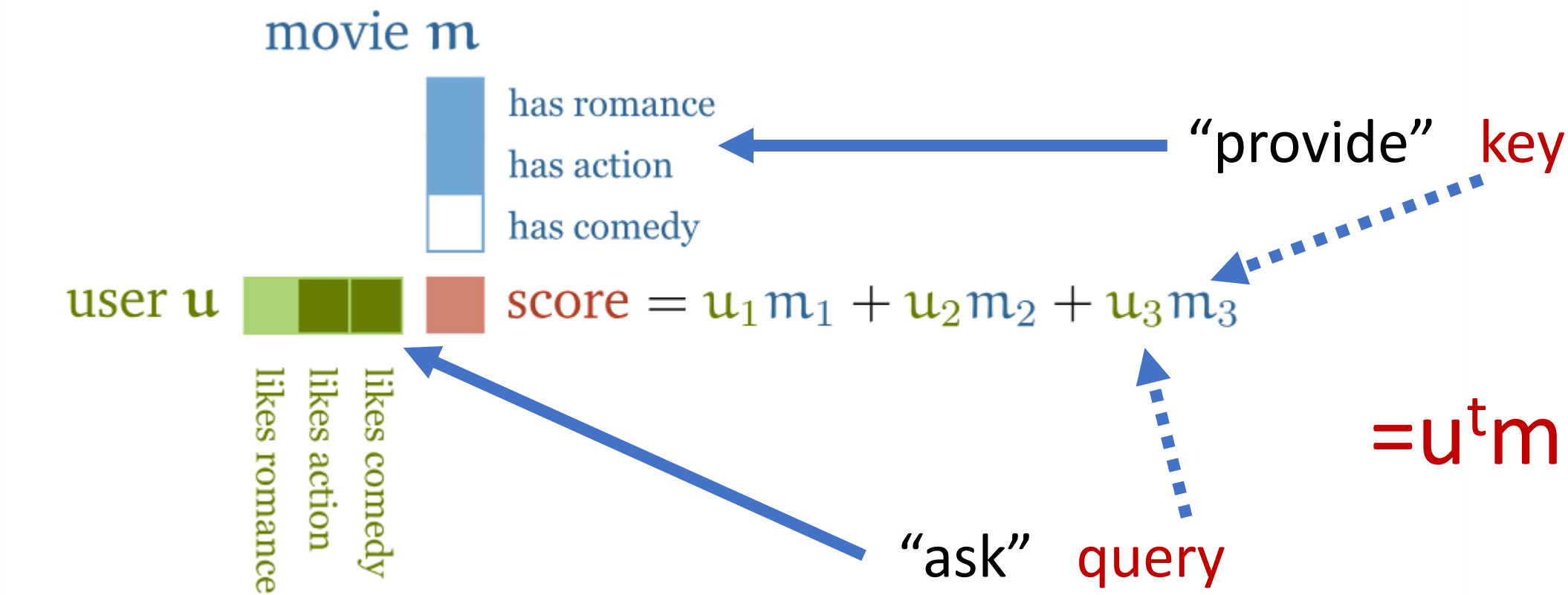
Understanding why self-attention works, in particular the dotprot construction

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

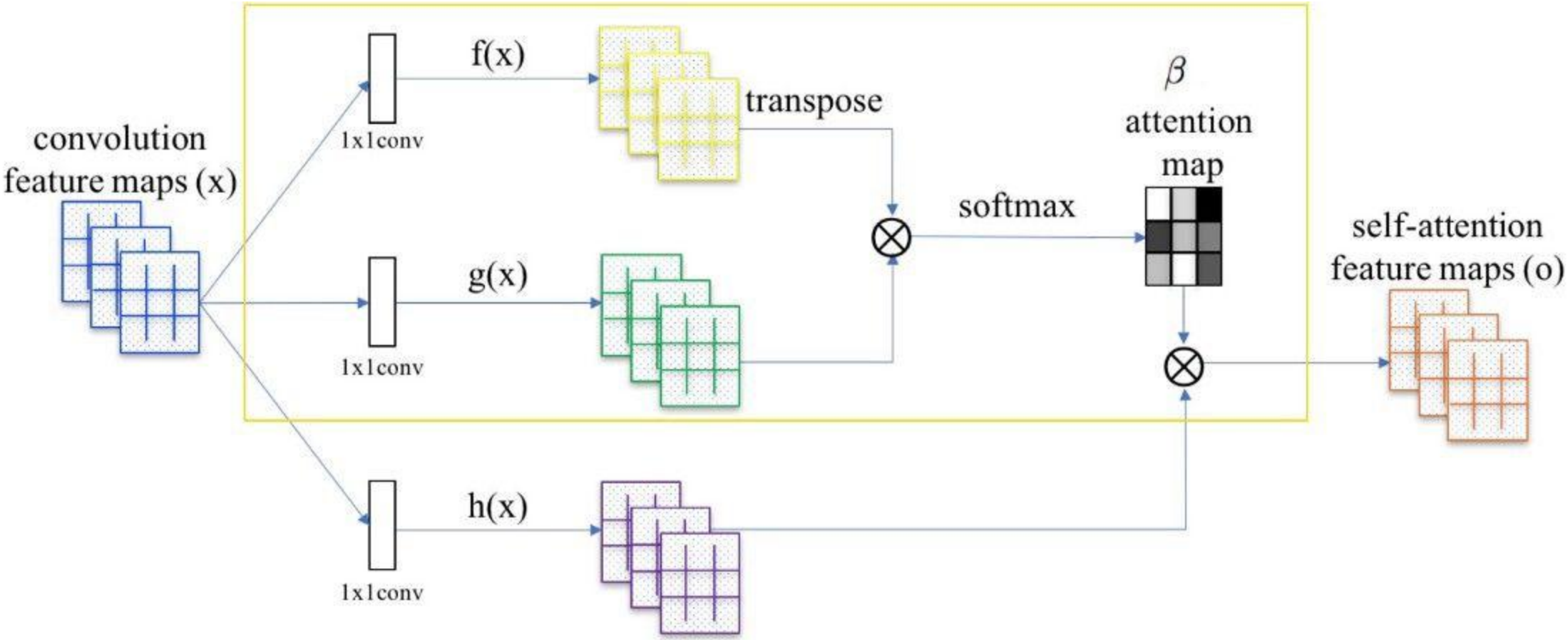


Understanding why self-attention works, in particular the dotprot consTruction

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



In practice, attention is a bit more complex



ATTENTION PAPER

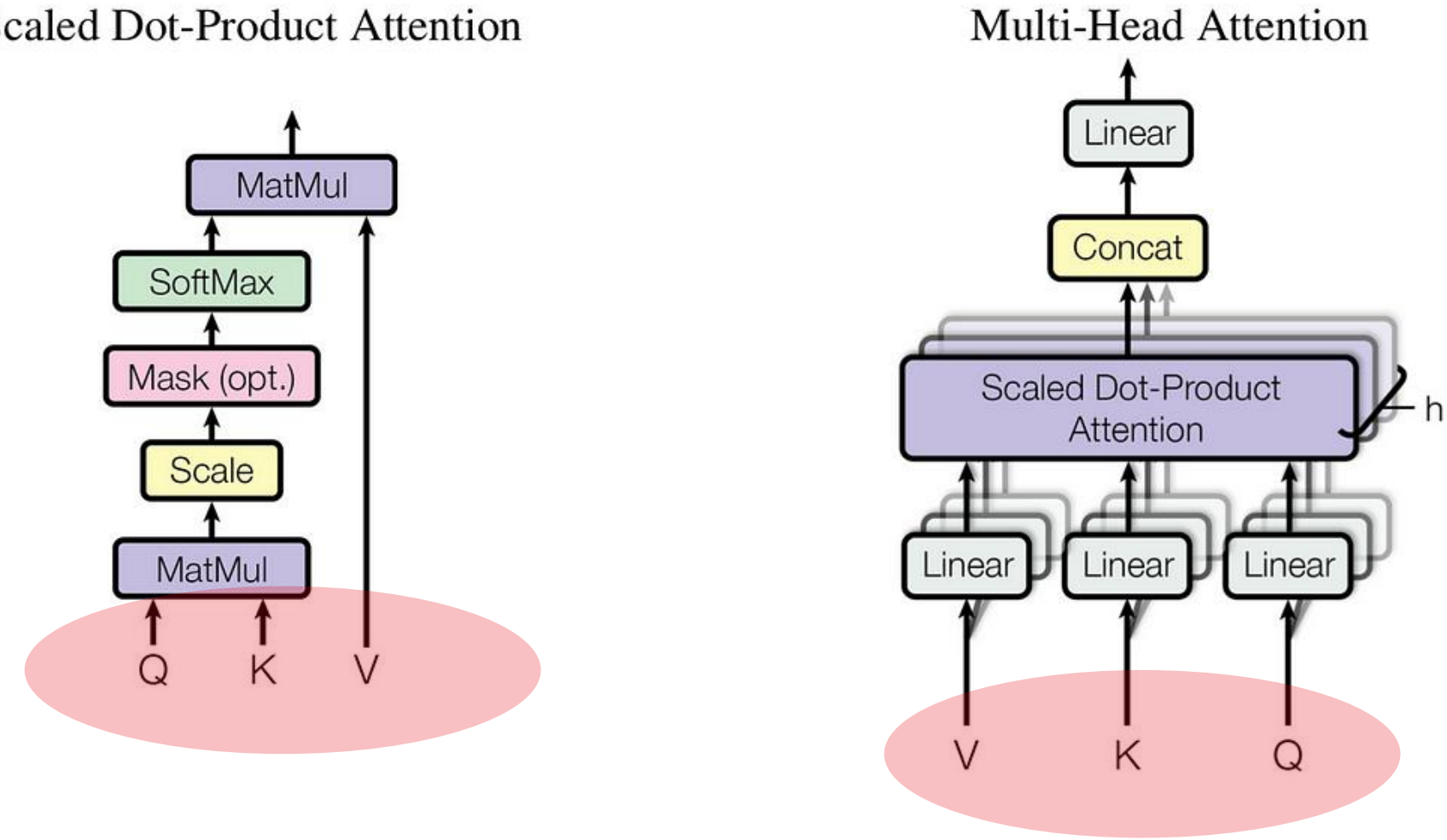


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

Core of the Transformer is Self Attention as in the original paper: "**Attention** is all you need".

arxiv.org/pdf/1706.03762.pdf

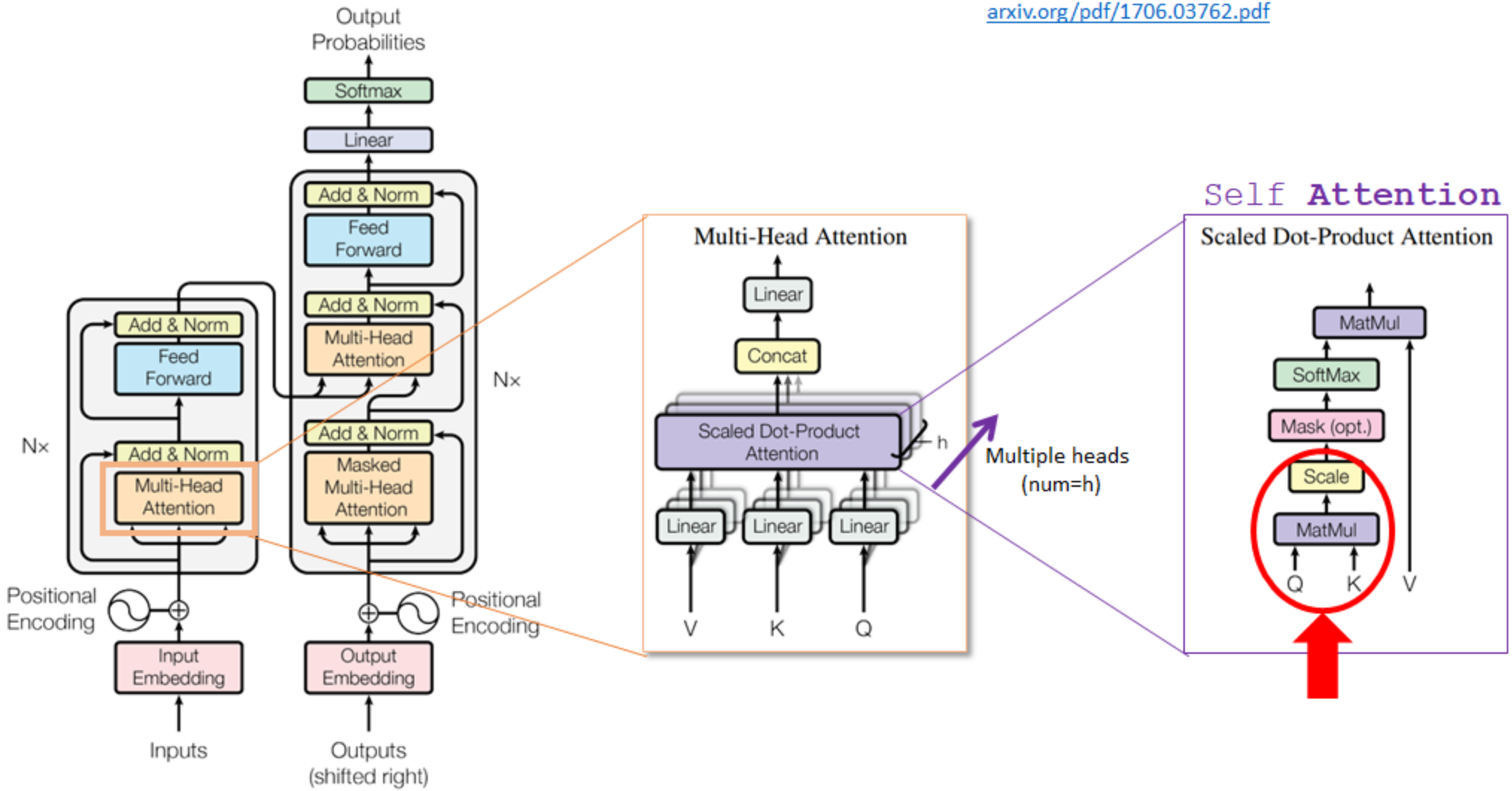


Figure 1: The Transformer - model architecture.

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 by[1]

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

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

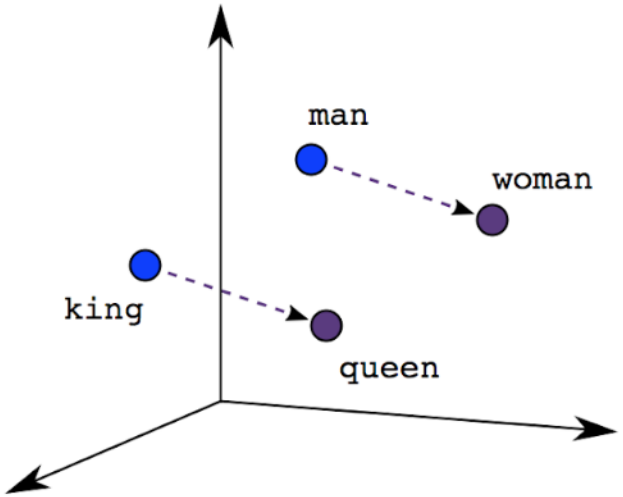
$$\text{MultiHeadAttn}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Concat}(\text{head}_1,\dots,\text{head}_H)\mathbf{W}^O, \tag{2}$$

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

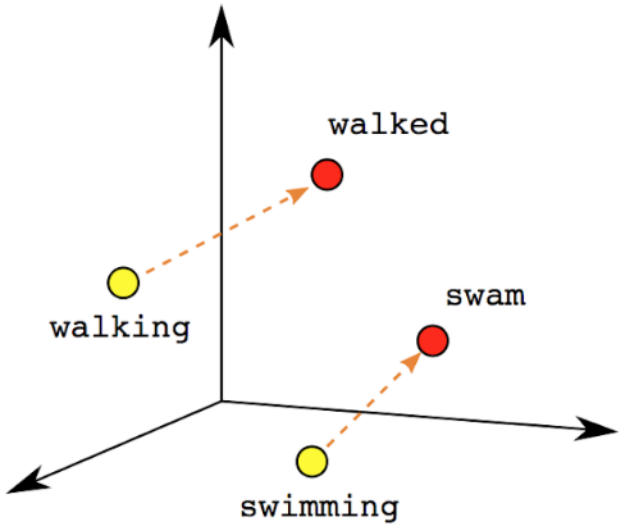
In Transformer, there are three types of attention in terms of the source of queries and key–value pairs:

- *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]

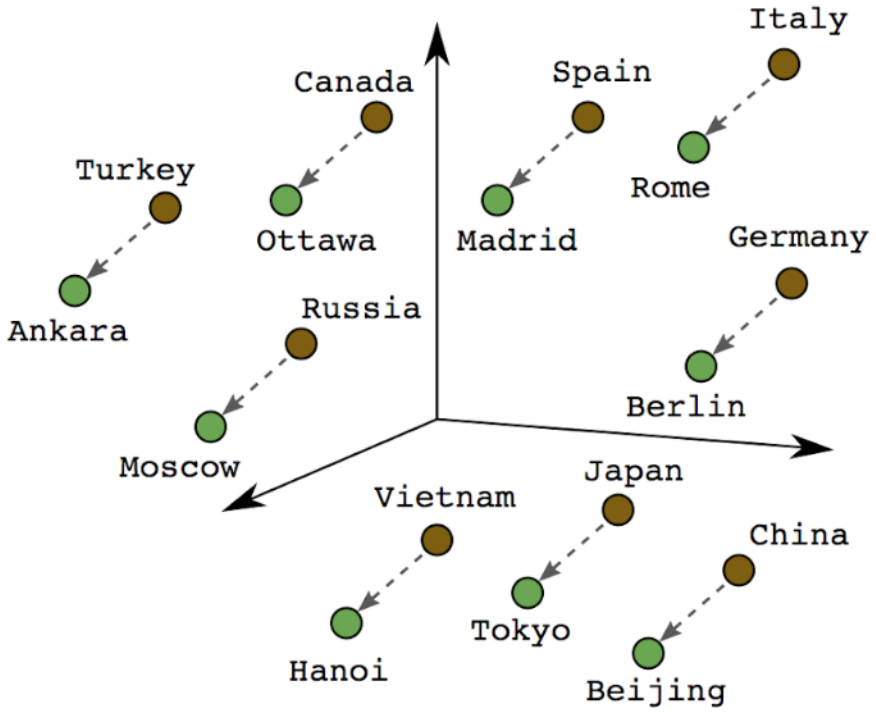
WORDS AS VECTORS (WORD EMBEDDINGS)



Male-Female



Verb Tense



Country-Capital

Dimension??

100, 300 (word2vec)

768 (BERT, …)

1024 (Wav2vec2.0)

Intro

- We use Transformers every day

- Transformer-based models are topic of investigation in multiple research directions, such as

- Explainability (interpretabilty – goes further)
- Approximation theory
- Linguistics

Intro

- Explainability
  - addressing the question to what extent decisions by Deep Neural Networks (DNNs) can be ’explained’ and understood.
  - The ’classical’ techniques (probing, feature attribution, integrated gradients, LIME, SHAP) appear often less useful than hoped in Transformer-based models
  - Mechanistic Interpretability is a novel and expanding group of techniques (’computational lenses’) with the goal to understand which decisions are made in the network, to aid formal verification and human interpretability.

LIME and SHAP

LIME Local Interpretable Model-agnostic Explanations

- approximates the model locally with a simpler model to explain predictions.
- fast, approximate, local, ignorant

SHAP SHapley Addictive exPlanations

- fairly distributes credit for a prediction among features using game theory (shapley features)
- more mathematically grounded, consistent, but computationally more expensive

Intro

- Approximation theory
  - mathematical underpinning of so-called ’limited width/depth networks’.
  - a classical result says that (under specific conditions) feedforward networks with sufficient width and depth are general approximators.
  - Approximation theorems say what neural networks can learn in terms of mappings

Current speech encoder/decoder architectures can learn highly complex mappings from the audio feature space to the word embedding space, due to stacked non-linear layers and weighted context dependency.

Intro

- Linguistics
  - In formal linguistics, sentences obey a particular structure: as a linear sequence of phrases, or of nested phrases
  - Formally, grammatical complexity can be organized along a scale inspired by the Chomsky hierarchy (Finite, Regular, Context-Free, mildly Context-Sensitive, Context Sensitive, ...).
- In spoken language, and certainly in spontaneous speech, the structure of utterances may deviate substantially from the formal properties of sentences in written text, but spoken utterances are not random and still obey a grammatical structure

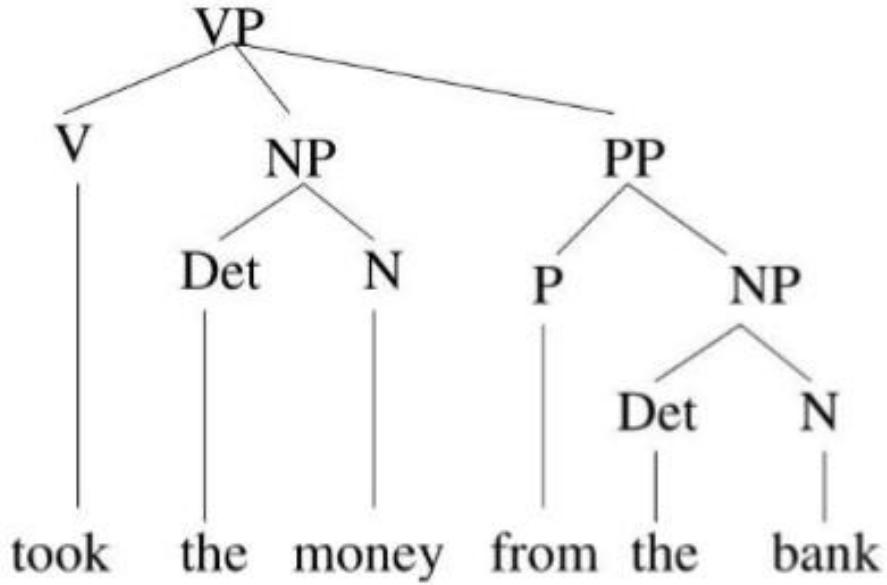
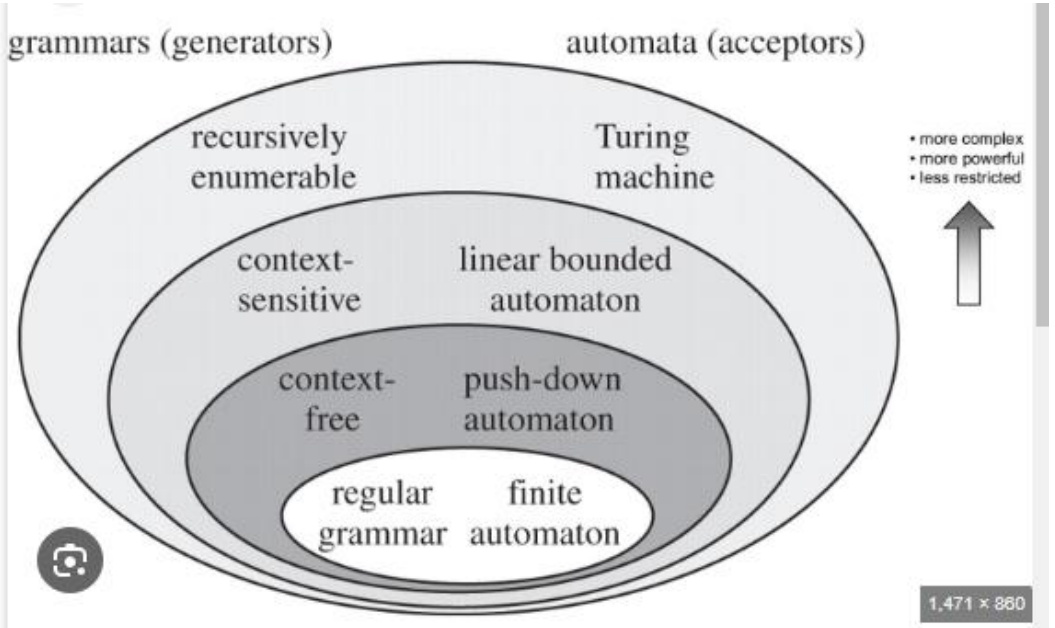


Figure 1: *Example of the phrase structure of a sentence.*



The hierarchy [edit]

The following table summarizes each of Chomsky's four types of grammars, the class of language it generates, the type of automaton that recognizes it, and the form its rules must have. The classes are defined by the constraints on the productions rules.

Grammar	Languages	Recognizing Automaton	Production rules (constraints)*	Examples[5][6]
Type-3	Regular	Finite-state automaton	$A \to \mathrm{a}$ $A \to \mathrm{a}B$ (right regular) or $A \to \mathrm{a}$ $A \to B\mathrm{a}$ (left regular)	$L = \{a^n \mid n > 0\}$
Type-2	Context-free	Non-deterministic pushdown automaton	$A \to \alpha$	$L = \{a^n b^n \mid n > 0\}$
Type-1	Context-sensitive	Linear-bounded non-deterministic Turing machine	$\alpha A \beta \to \alpha \gamma \beta$	$L = \{a^n b^n c^n \mid n > 0\}$
Type-0	Recursively enumerable	Turing machine	$\gamma \to \alpha$ (γ non-empty)	$L = \{w \mid w$ describes a terminating Turing machine$\}$

* Meaning of symbols:
- a = terminal
- A, B = non-terminal
- α, β, γ = string of terminals and/or non-terminals

Transformers are powerful

leeuw olifant slang leeuw muis aap aap kat leeuw aap
leeuw slang slang slang kat leeuw olifant leeuw hond slang
aap hond kanarie leeuw slang slang olifant hond leeuw slang
kat leeuw hond kat olifant kanarie olifant leeuw kanarie slang
hond aap leeuw kanarie hond olifant leeuw kanarie kanarie hond
slang slang leeuw olifant muis leeuw muis kanarie kanarie kat
leeuw hond slang leeuw olifant aap aap slang leeuw aap
leeuw muis kat olifant kat kat aap leeuw slang kat
aap leeuw olifant kat hond hond muis slang hond leeuw
muis muis aap leeuw aap slang olifant leeuw kat leeuw
hond leeuw olifant olifant leeuw kat kanarie aap hond aap
kat leeuw hond hond kanarie kat kanarie kat olifant leeuw
hond muis aap slang leeuw kanarie olifant slang leeuw muis

+

hond aap slang olifant slang slang leeuw leeuw kat muis
kat kat aap kat kanarie muis muis aap olifant leeuw
aap olifant kanarie aap leeuw kat muis kat hond kanarie
kat aap kat kat olifant hond hond aap muis hond
slang kanarie slang kanarie aap leeuw aap kanarie leeuw aap
hond muis muis olifant kanarie olifant slang leeuw kanarie slang
slang kat muis muis slang slang muis leeuw slang slang
leeuw aap hond hond leeuw kanarie hond slang hond olifant
muis leeuw aap hond kanarie kanarie kanarie aap kat slang
olifant muis muis leeuw kanarie olifant kanarie kat leeuw slang
hond kat muis aap leeuw leeuw leeuw kat kanarie leeuw
hond slang aap slang muis slang leeuw kat slang olifant
muis muis leeuw leeuw leeuw aap kanarie aap hond kanarie

−