SPONSORED BY THE

Federal Ministry of Education and Research

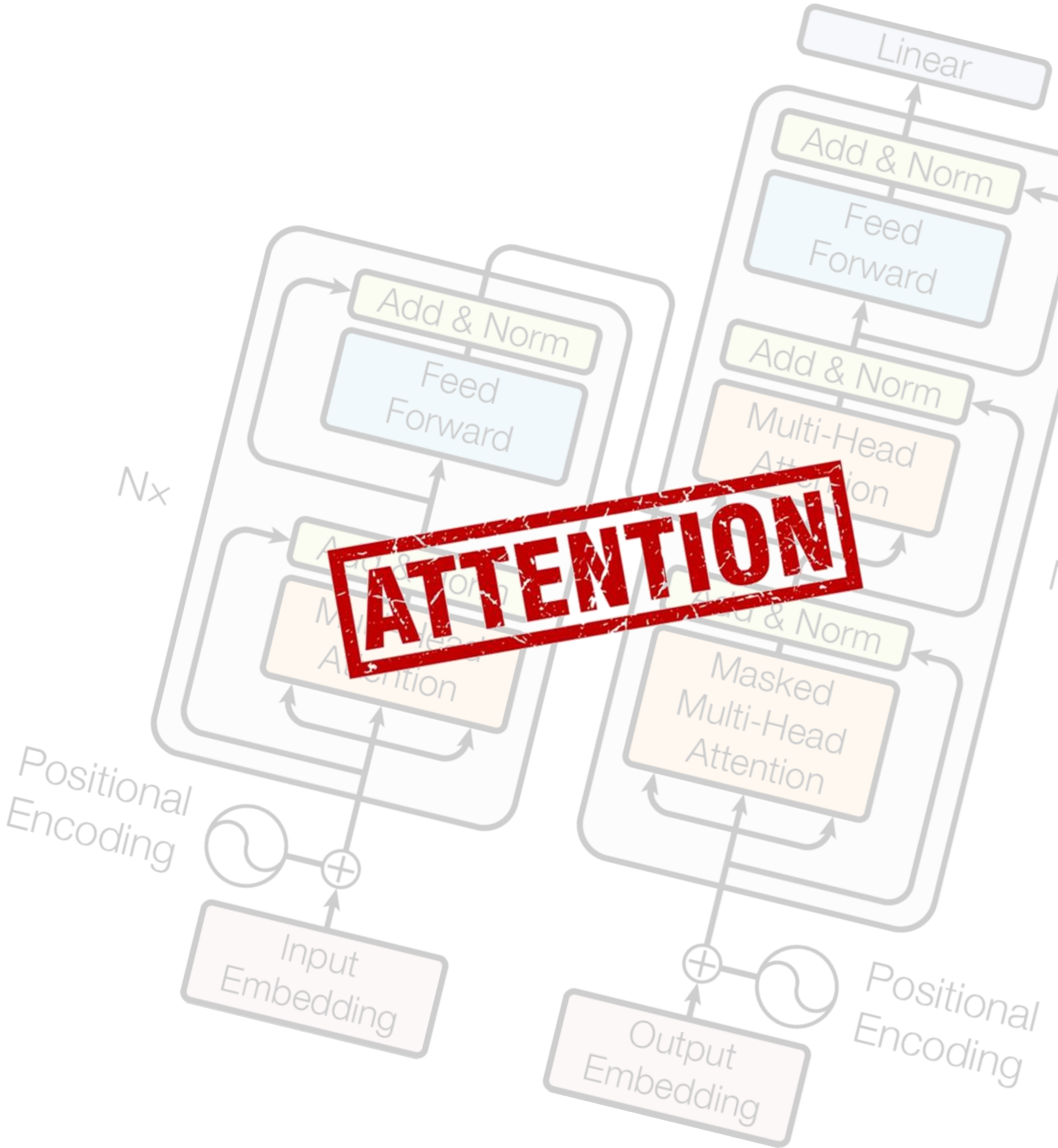
III. Physikalisches Institut A

RWTH AACHEN UNIVERSITY

Transformers

Active Training Course "Advanced Deep Learning"

ERUM DATA HUB



Niklas Langner
RWTH Aachen University

Current Hype: Text-to-Image Models

Text Prompt

Photography of an Astronaut wearing a green spacesuit standing in front of the Colosseum on the moon, with a bouquet of roses in his hand

Text Analysis
(Typically ***Transformer***)

Image Generation
(Typically *Diffusion*)

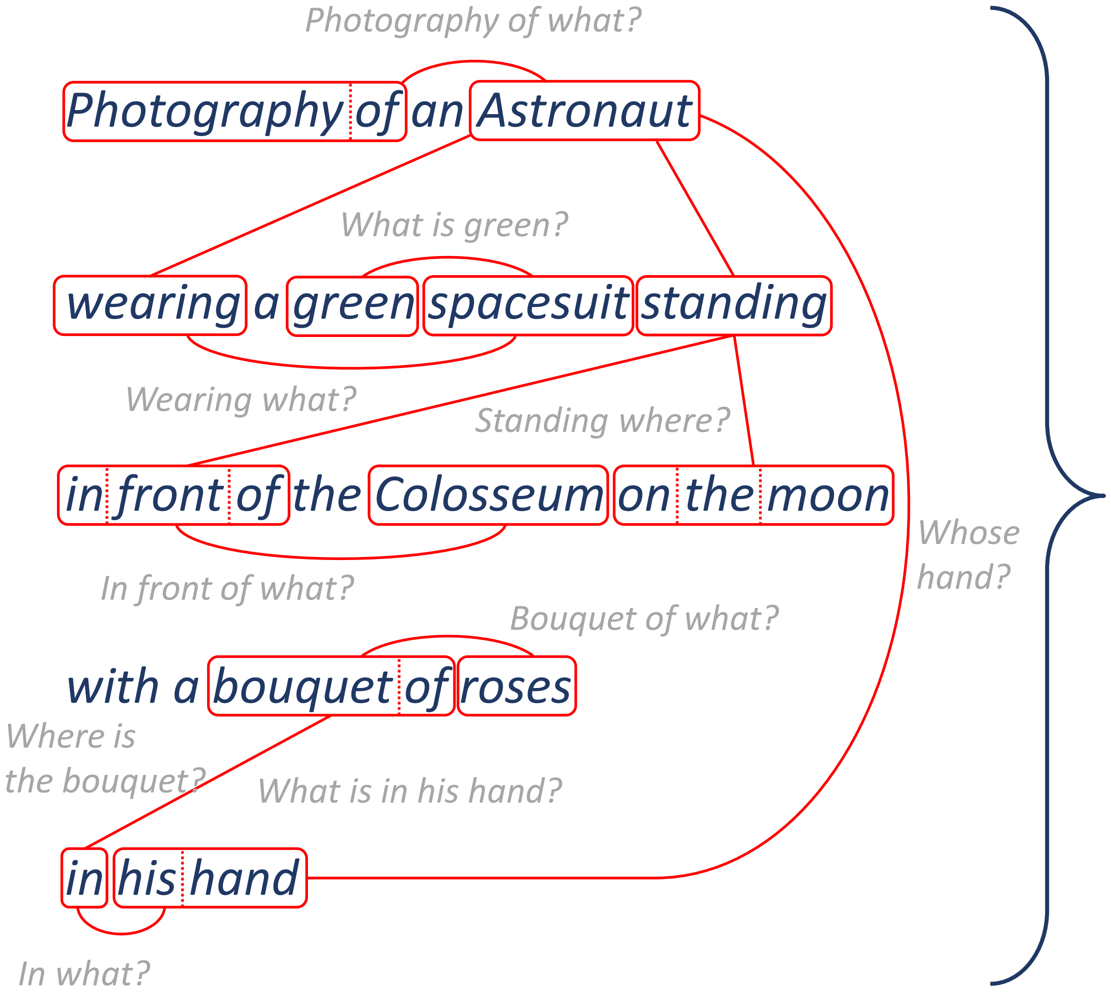
Unusual color,
unusual location,
unusual item

→ Network must **understand prompt** well!

Output Image

Generated with Stable Diffusion
https://arxiv.org/abs/2112.10752
https://github.com/CompVis/stable-diffusion

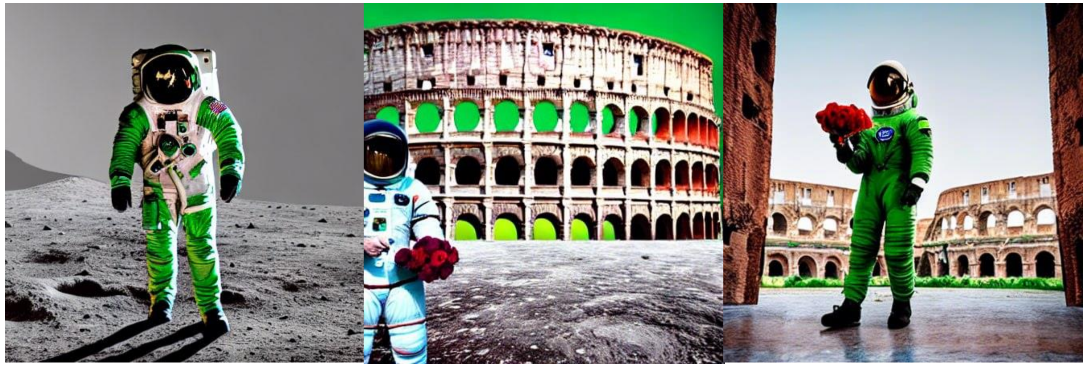
Understanding the Text Prompt



Cherry-picked examples:

Successful generations:

Misunderstandings:



https://github.com/CompVis/stable-diffusion

Understanding a long text is **hard**, but network (often) manages to do it → **How does it work?**

Natural Language Processing

Example: Sequence to sequence translation

Attention is all you need

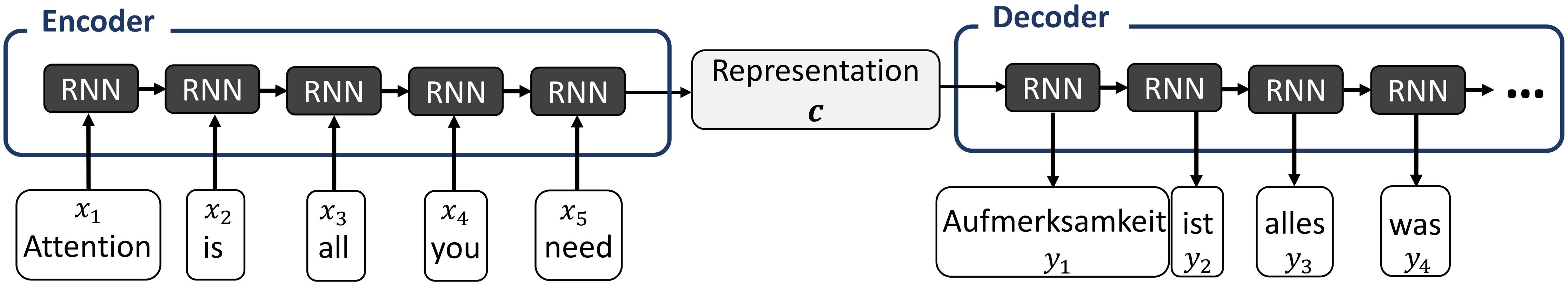
$\boldsymbol{x} = \{x_1, x_2, x_3, x_4, x_5\}$ → $\boldsymbol{y} = \{y_1, y_2, y_3, y_4, y_5, y_6\}$

Aufmerksamkeit ist alles, was du brauchst

Inputs and outputs: Embedding vectors corresponding to specific words

Before transformers: Recurrent Neural Networks („RNNs“, e.g. **LSTMs**)

Approach: Analyze the data **sequentially** → current step always depends on all previous steps



niklas.langner@rwth-aachen.de

Sequential Data Processing

Sequential analysis is helpful to understand **context**!

I like your house, **it** is great! → Ich mag dein Haus, **es** ist toll!

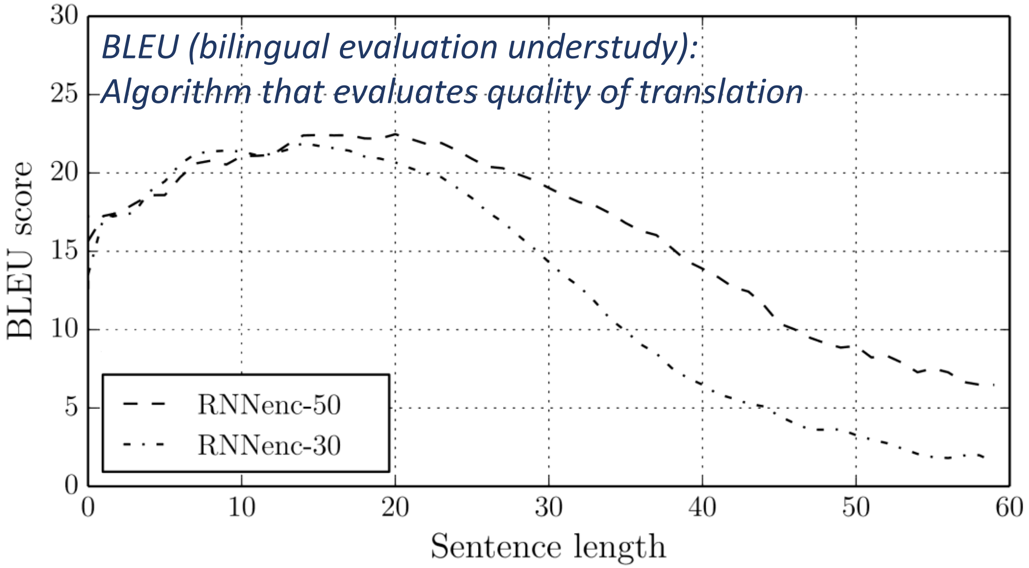
I like your dog, **it** is great! → Ich mag deinen Hund, **er** ist toll!

I like your cat, **it** is great! → Ich mag deine Katze, **sie** ist toll!

Correct translations of words can depend on **previous** or **following** words

In theory: Using an RNN, current step has access to information from **all previous steps**

In practice: Only works well for **small sequences** (<20 timesteps) due to too small gradients in regard to far away timesteps

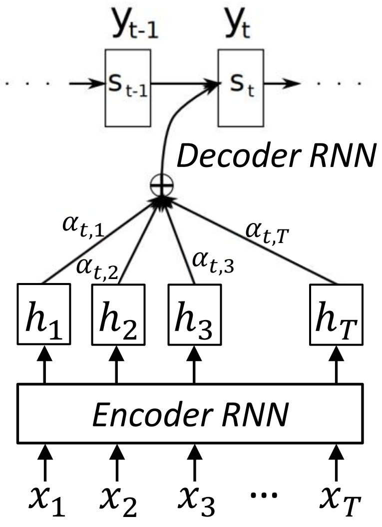


https://arxiv.org/abs/1409.0473

Solution: Attention

Use intermediate results of RNN layers and **combine them using weights** α (*alignment score)*

Attention Example



Nonlinear function e.g. LSTM

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j$$

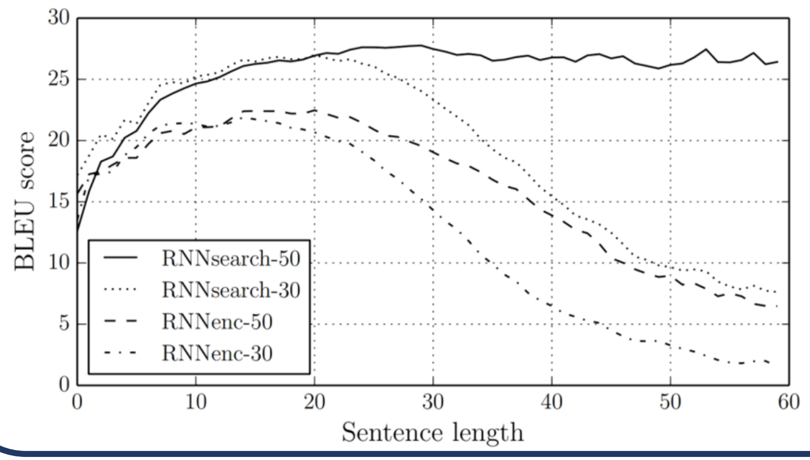
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

softmax

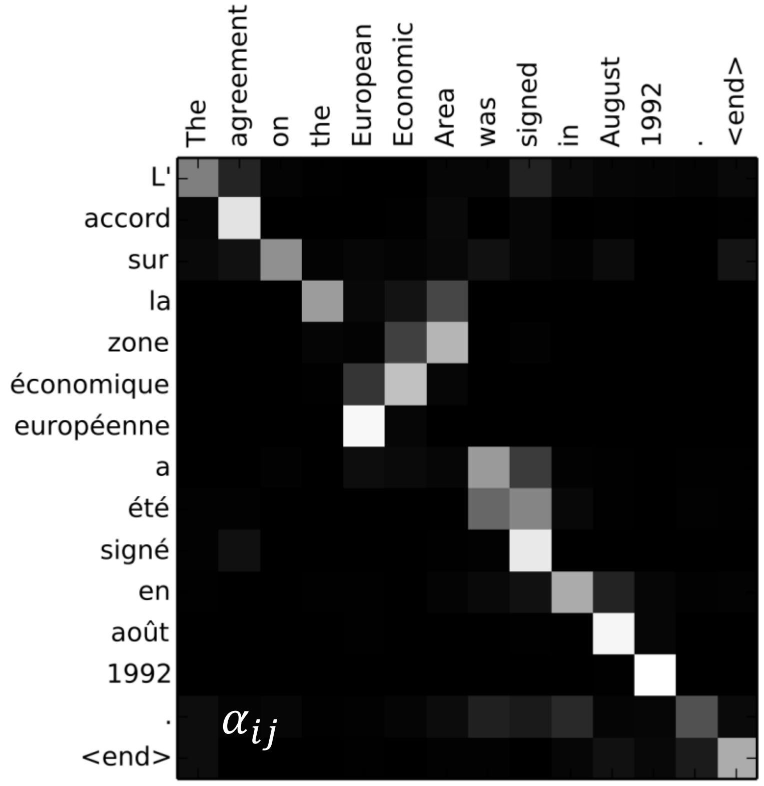
$$e_{ij} = \text{score}(s_{i-1}, h_j)$$

Alignment Score Function

Different for each step!



Greatly improves performance translating **long sentences**



https://arxiv.org/abs/1409.0473

Attention Mechanisms

$$s_i = f(s_{i-1}, y_{i-1}, c_i) \qquad c_i = \sum_{j=1}^{T} \alpha_{ij} h_j \qquad \alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

$$e_{ij} = \text{score}(s_{i-1}, h_j)$$

Alignment score: Many different options

Commonly used neural network approach → Additive

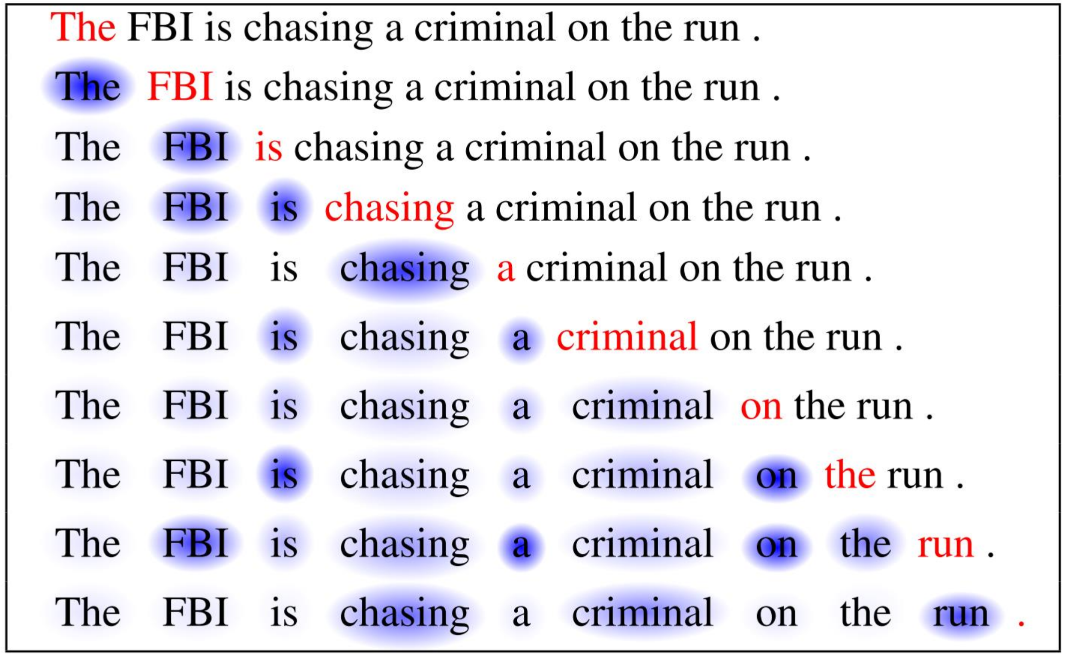
Name	Alignment score function	Citation
Content-base attention	$\text{score}(s_{i-1}, h_i) = \cos(\theta)$ where θ is the angle between s_{i-1} and h_j	Graves2014
Additive	$\text{score}(s_{i-1}, h_j) = v_a^{\mathrm{T}} \tanh(W_a[s_{i-1}; h_j])$	Bahdanau2015
Location-Based	$\alpha_{ij} = \text{softmax}(W_a s_i)_{\mathrm{j}}$ *Note: This simplifies the softmax alignment to only depend on the target position.*	Luong2015
General	$\text{score}(s_{i-1}, h_j) = s_{i-1}^{\mathrm{T}} W_a h_j$ *where W_a is a trainable weight matrix in the attention layer.*	Luong2015
Dot-Product	$\text{score}(s_{i-1}, h_j) = s_{i-1}^{\mathrm{T}} h_j$	Luong2015
Scaled Dot-Product	$\text{score}(s_{i-1}, h_j) = \frac{s_{i-1}^{\mathrm{T}} h_j}{\sqrt{n}}$ *Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.*	Vaswani2017

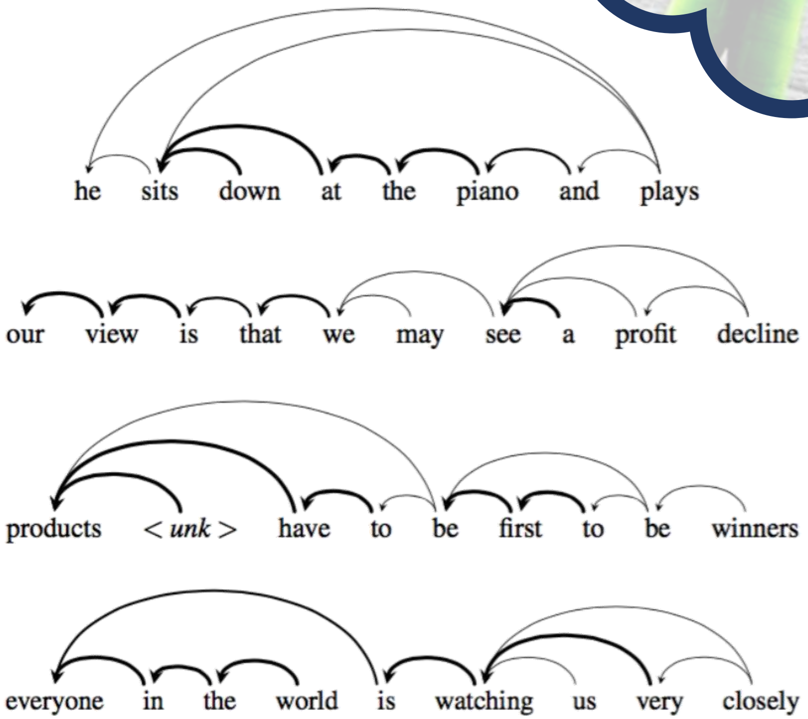
Important for Transformer (↑ Scaled Dot-Product)

https://lilianweng.github.io/posts/2018-06-24-attention/

Self-Attention

Relate input **to itself** to determine **self-attention:**





Bolder line: **Higher** attention

⇒ Adding **attention** mechanisms to RNNs **improves the performance** (especially for **long sentences**) and can enable **interesting insights**

https://arxiv.org/abs/1601.06733

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

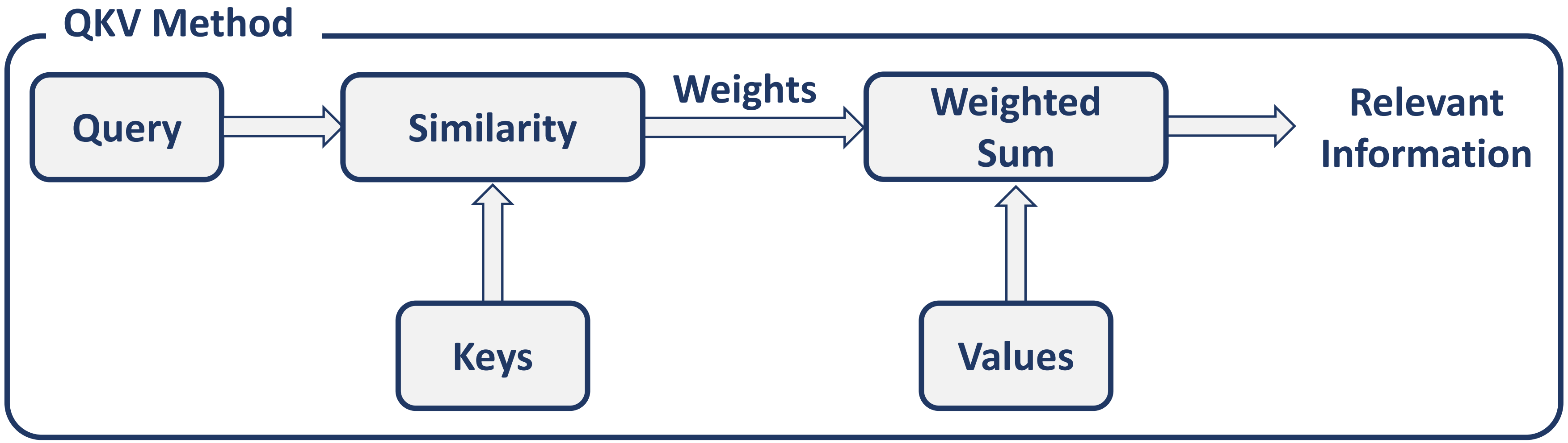
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to

Introducing: The Transformer

https://arxiv.org/abs/1706.03762

Transformer: Attention Mechanism

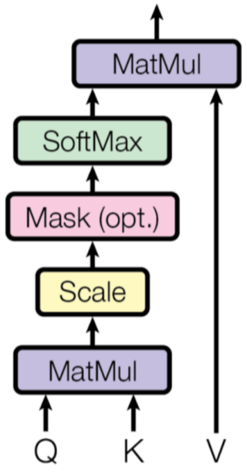
Interpret attention as function of **query Q**, **keys K** and **values V**



Scaled dot-product attention:

Dot-product (vector projection) as **similarity measure**

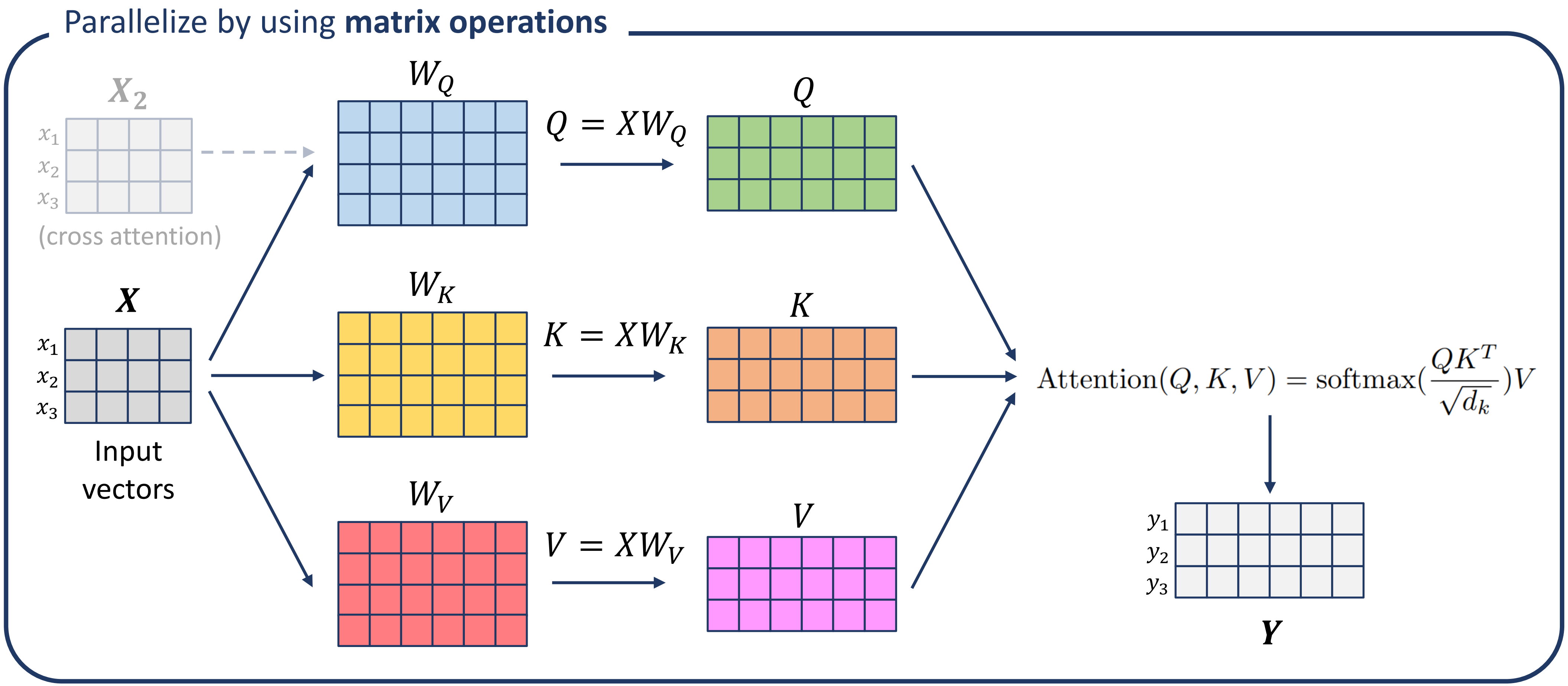
$$\mathrm{score}(s_{i-1}, h_j) = \frac{s_{i-1}^{\mathrm{T}} h_j}{\sqrt{n}} \Longrightarrow \mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



d_k: length of keys vector

https://arxiv.org/abs/1706.03762

QKV-Self-Attention Implementation

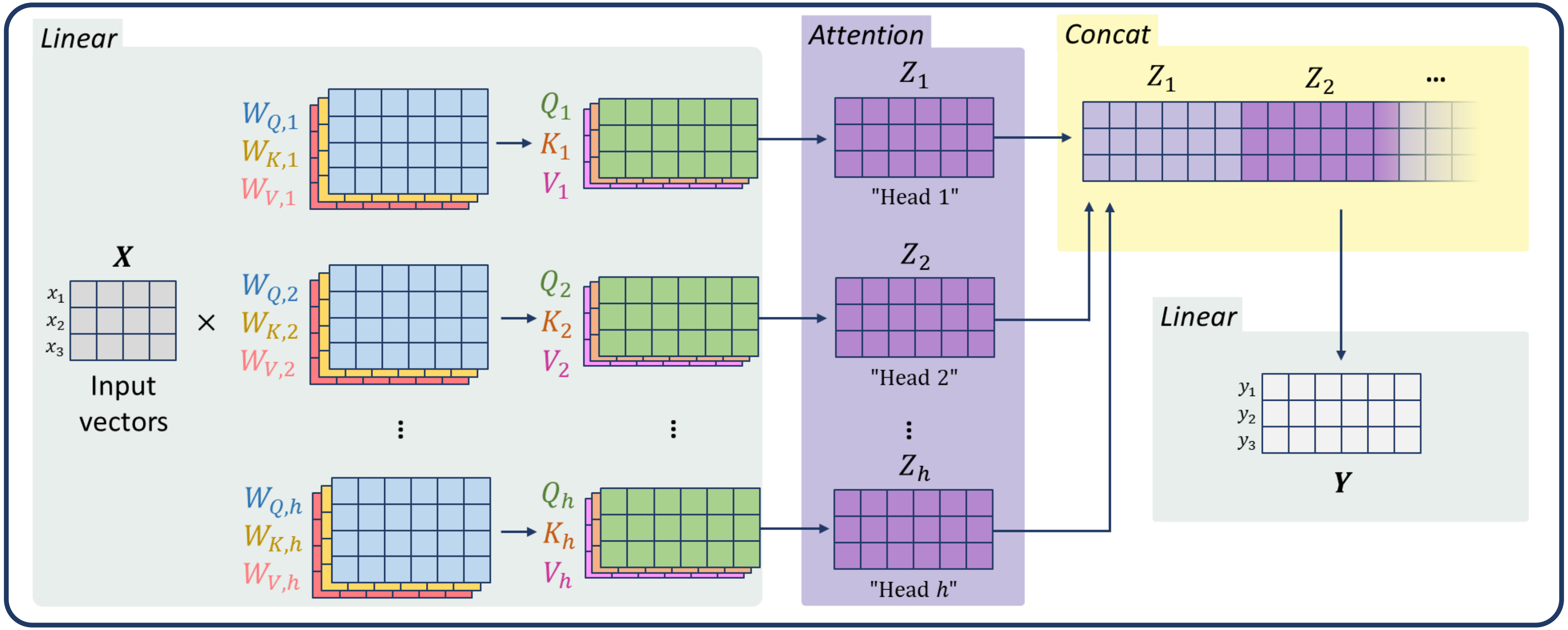


Here: Always use X → calculate **self attention**

To **relate X** to **different input data $\mathbf{X_2}$**, use X_2 to calculate Q (**cross attention**)

https://arxiv.org/abs/1706.03762

Multi-Head Attention



Heads are **independent** of each other!

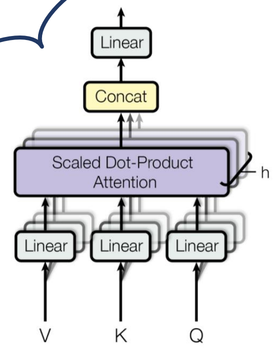
→ Can be **trained in parallel** on multiple GPUs

→ Enables **training on huge** datasets in reasonable time

Input treated as set instead of sequence → permutation invariant ⚡

→ Use **positional encoding**!

Building block of transformer

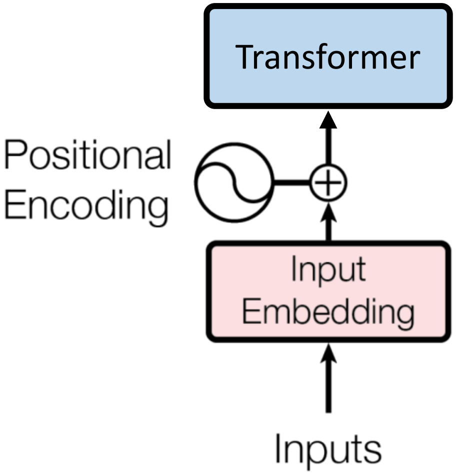


https://arxiv.org/abs/1706.03762

Positional Encoding

Positional Encoding PE is **added to embedded vectors from inputs** to pass positional information to transformer

Many different options (both **trainable** and **fixed**), paper uses **fixed encoding**:

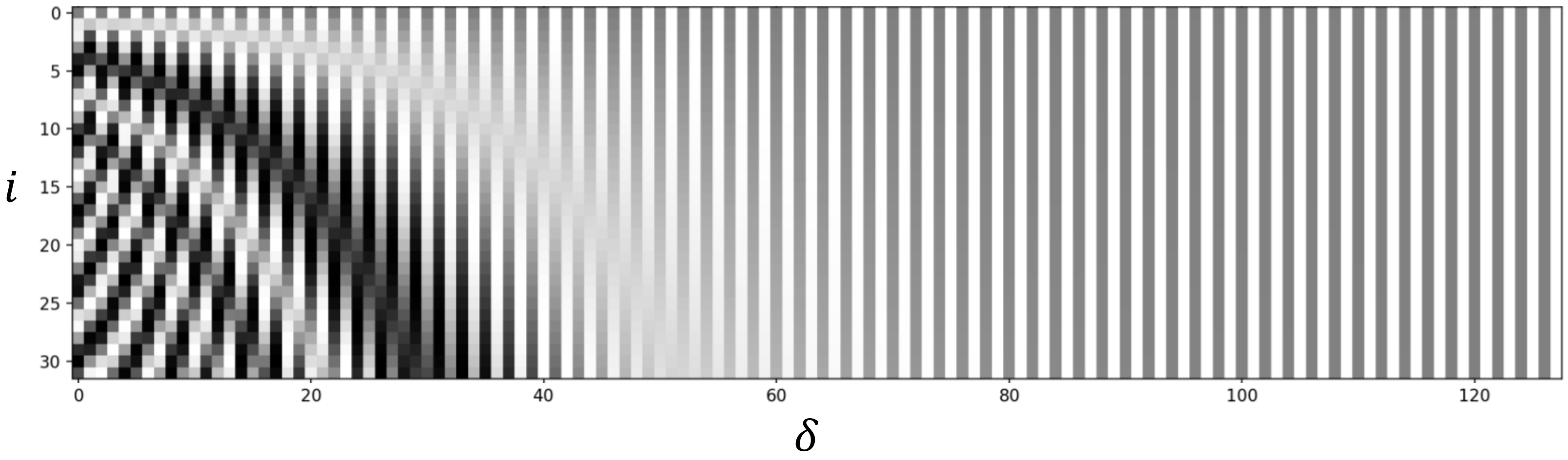


Example

$$\mathrm{PE}(i,\delta) = \begin{cases} \sin\left(\frac{i}{10000^{2\delta'/d}}\right) & \text{if } \delta = 2\delta' \\ \cos\left(\frac{i}{10000^{2\delta'/d}}\right) & \text{if } \delta = 2\delta' + 1 \end{cases}$$

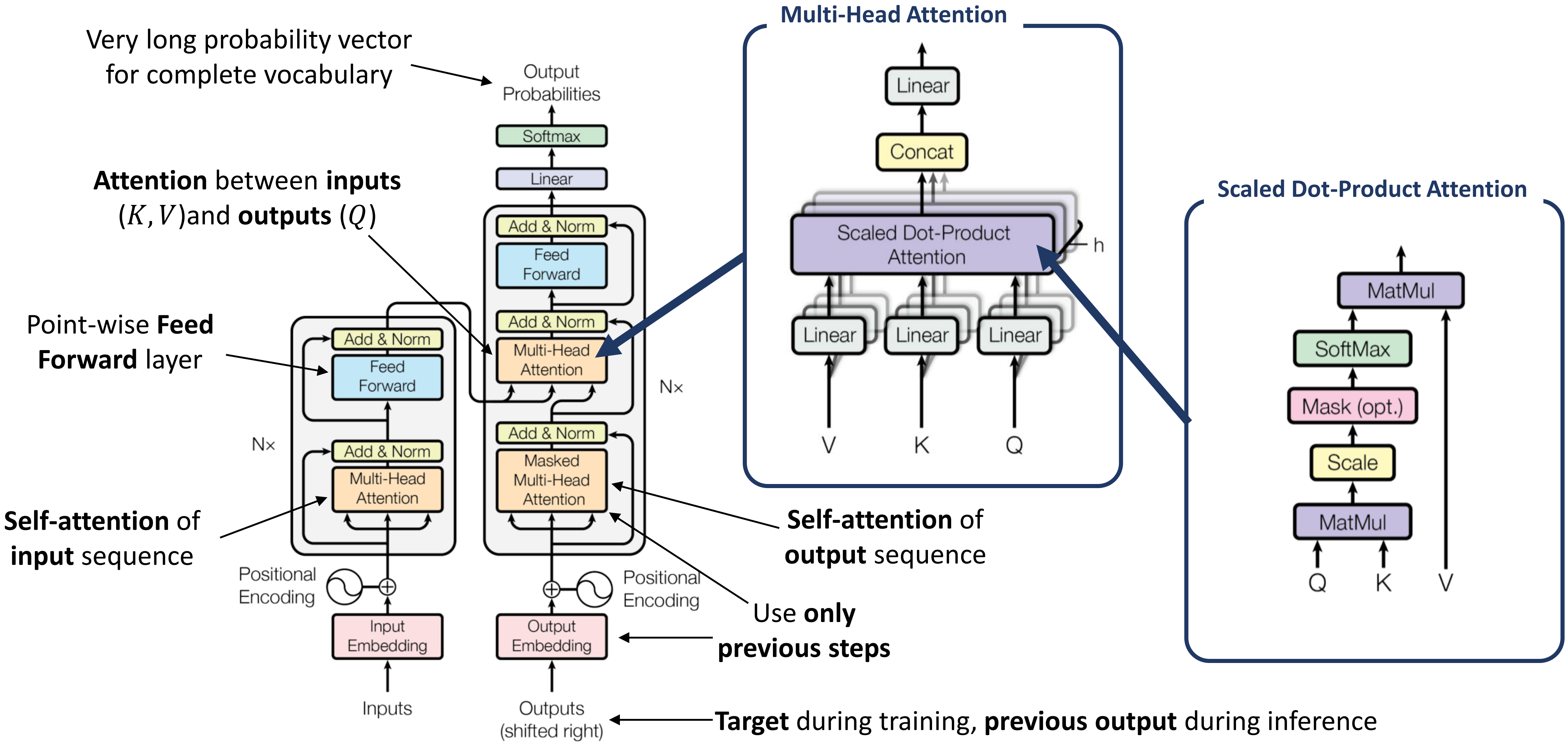
i: word position
δ: vector dimension
d: vector length in network

https://lilianweng.github.io/posts/2020-04-07-the-transformer-family/



https://arxiv.org/abs/1706.03762

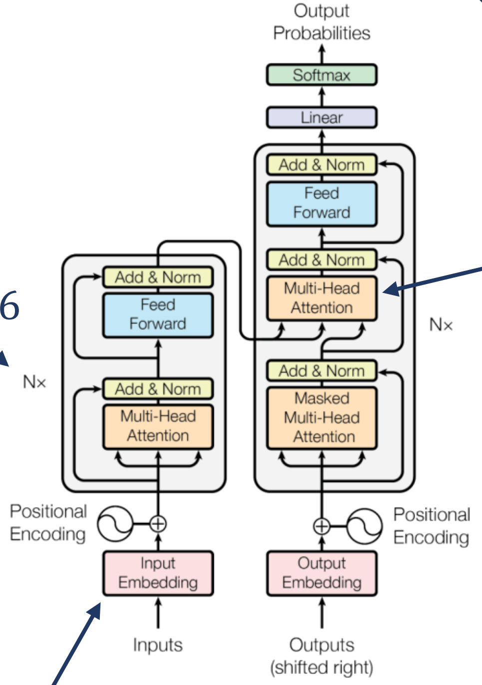
Full Transformer Architecture



https://arxiv.org/abs/1706.03762

Text Translation with Transformer

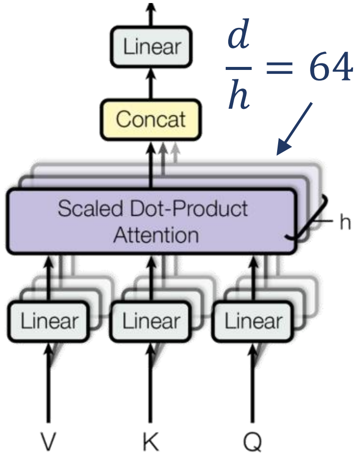
Transformer



$N = 6$

Dimensionality $d = 512$

Multi-Head Attention



$\frac{d}{h} = 64$ dimensional

$h = 8$ independent heads

→ can be **trained on 8 GPUs simultaneously**
Total computational cost is similar to that of single-head attention with full dimensionality of 512

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

https://arxiv.org/abs/1706.03762

Impact of Transformers – Language Processing

GLUE https://gluebenchmark.com/leaderboard

Rank	Name	Model	URL	Score
1	Microsoft Alexander v-team	Turing ULR v6	↗	91.3
2	JDExplore d-team	Vega v1		91.3
3	Microsoft Alexander v-team	Turing NLR v5	↗	91.2
4	DIRL Team	DeBERTa + CLEVER		91.1
5	ERNIE Team - Baidu	ERNIE	↗	91.1
6	AliceMind & DIRL	StructBERT + CLEVER	↗	91.0
7	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4	↗	90.8
8	HFL iFLYTEK	MacALBERT + DKM		90.7
9	PING-AN Omni-Sinitic	ALBERT + DAAF + NAS		90.6
10	T5 Team - Google	T5	↗	90.3
11	Microsoft D365 AI & MSR AI & GATECH	MT-DNN-SMART	↗	89.9
12	Huawei Noah's Ark Lab	NEZHA-Large		89.8
⋮				
79	GLUE Baselines	BiLSTM+ELMo+Attn	↗	70.0

All using Transformers (ranks 1–12)

Transformers beat LSTMs/Recurrent Neural Networks as state of the art in natural language processing!

Huge transformer models are widely-used for language processing:

- **Google BERT**
  (340 Million parameters)
- **Google T5**
  (11 billion parameters)
- **OpenAI GPT-3**
  (175 billion parameters)
- **Microsoft Megatron-Turing NLG**
  (530 billion parameters)

→ Models using more than one **trillion** parameters are already tested

Transformer for Language Processing - Overview

- **Transformers**: Use **attention-mechanism** to analyze data → **Relevance** between different inputs
- Positional Encoding enables the **analysis of sequences**
  → Ability to understand **long and complicated** texts
  - **Transformers replaced RNNs as state-of-the-art methods for natural language processing**

Enabled new applications:

Generating large texts

A robot wrote this entire article. Are you scared yet, human?
GPT-3

We asked GPT-3, OpenAI's powerful new language generator, to write an essay for us from scratch. The assignment? To convince us robots come in peace

Understanding complex prompts

Writing code

Your AI pair programmer

GitHub Copilot uses the OpenAI Codex to suggest code and entire functions in real-time, right from your editor.

Conversational models

Hi! I am Mount Everest. What would you like to know about me?

Why do people climb you?

Some people want to show others they could do it, others enjoy the hike and the scenery, and then some people just want to see the world.

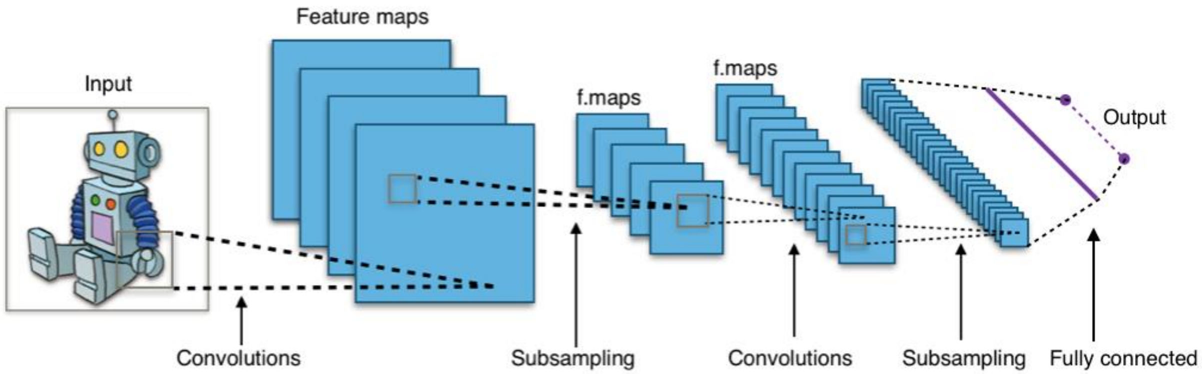
Who was the first?

Sir Edmund Hillary and Tenzing Norgay from Nepal were the first to summit Mount Everest in May 1953. They were joined by several others at different times in the successful attempt.

…and many more

Image Recognition

Previous state of the art in image recognition: **Convolutional Neural Networks**

[1] 

- Learned **filters** that are used to scan images for features
- Inherently uses the **symmetries of 2D images**, i.e. which pixels are neighbors, where does the image end, etc.
- However: Can be **computationally demanding**

*"Looking forward to the next generation of **scalable vision models**, one might ask whether this domain-specific design is necessary, or if one could successfully leverage **more domain agnostic and computationally efficient architectures** to achieve state-of-the-art results."*

https://ai.googleblog.com/2020/12/transformers-for-image-recognition-at.html

⇒ **Vision Transformer**

[1] https://commons.wikimedia.org/wiki/File:Typical_cnn.png

Vision Transformer (ViT)

Ansatz

“In model design we **follow the original Transformer** (Vaswani et al., 2017) **as closely as possible**. An advantage of this intentionally simple setup is that scalable NLP Transformer architectures – and their efficient implementations – can be used almost out of the box.“

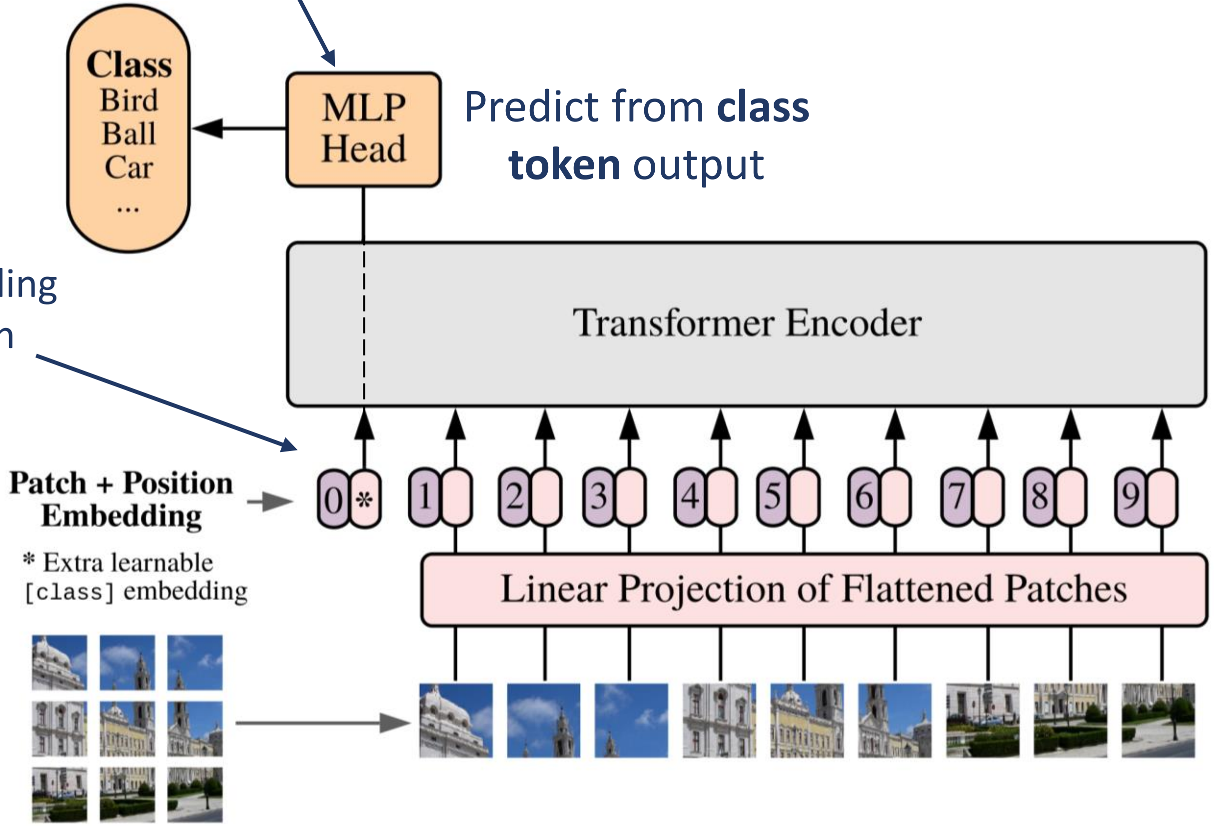
MLP for **final prediction**

Predict from **class token** output

Multihead Attention
Transformer from language processing task

Learnable positional encoding lets network know which patch is which

Split image into ***patches***



Flatten patches (**losing 2D structure**) and **project linearly to embedding** dimension (can also be done via **CNN in hybrid approach**)

https://arxiv.org/abs/2010.11929

Vision Transformer Performance

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

	Transformers			CNNs	
	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** $\pm$ 0.04	87.76 $\pm$ 0.03	85.30 $\pm$ 0.02	87.54 $\pm$ 0.02	88.4/88.5*
ImageNet ReaL	**90.72** $\pm$ 0.05	90.54 $\pm$ 0.03	88.62 $\pm$ 0.05	90.54	90.55
CIFAR-10	**99.50** $\pm$ 0.06	99.42 $\pm$ 0.03	99.15 $\pm$ 0.03	99.37 $\pm$ 0.06	–
CIFAR-100	**94.55** $\pm$ 0.04	93.90 $\pm$ 0.05	93.25 $\pm$ 0.05	93.51 $\pm$ 0.08	–
Oxford-IIIT Pets	**97.56** $\pm$ 0.03	97.32 $\pm$ 0.11	94.67 $\pm$ 0.15	96.62 $\pm$ 0.23	–
Oxford Flowers-102	99.68 $\pm$ 0.02	**99.74** $\pm$ 0.00	99.61 $\pm$ 0.02	99.63 $\pm$ 0.03	–
VTAB (19 tasks)	**77.63** $\pm$ 0.23	76.28 $\pm$ 0.46	72.72 $\pm$ 0.21	76.29 $\pm$ 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Transformer: More efficient

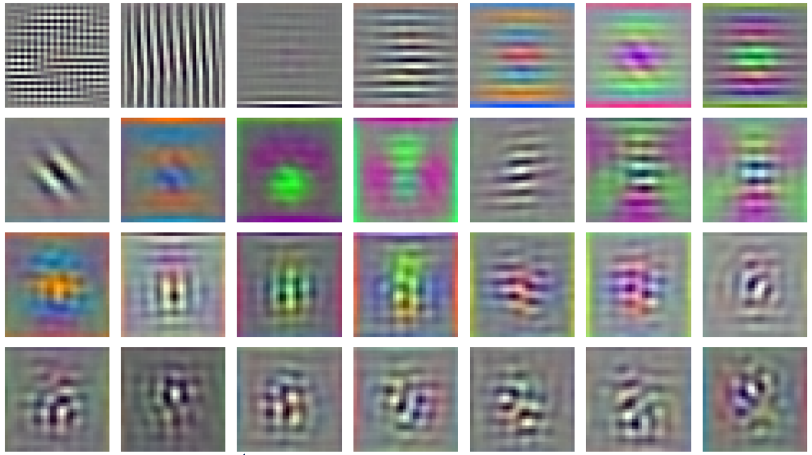
"ViT performs **significantly worse** than the CNN equivalent (BiT) when **trained on ImageNet** (1M images). However, on **ImageNet-21k** (14M images) performance is **comparable**, and on **JFT** (300M images), **ViT now outperforms BiT**."

⇒ **Vision Transformer** already **outperforms (highly-optimized) CNNs**, despite using architecture created for language processing

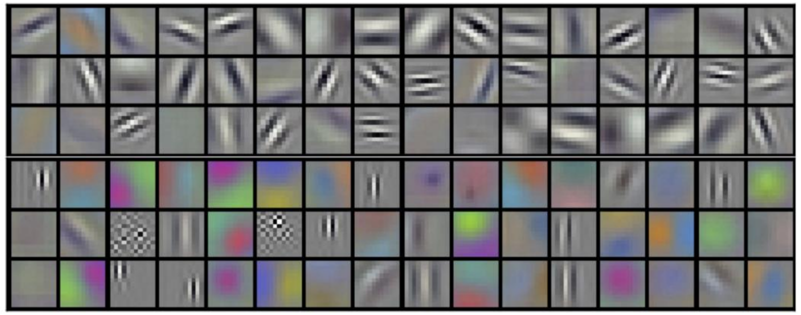
https://arxiv.org/abs/2010.11929

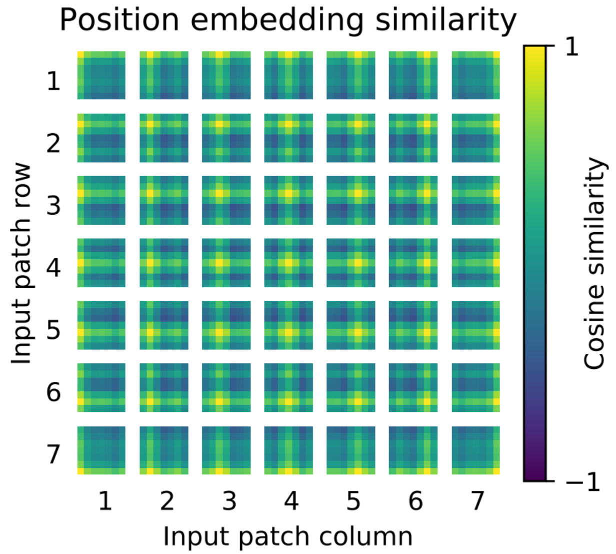
Vision Transformer Insight

RGB embedding filters
(first 28 principal components)

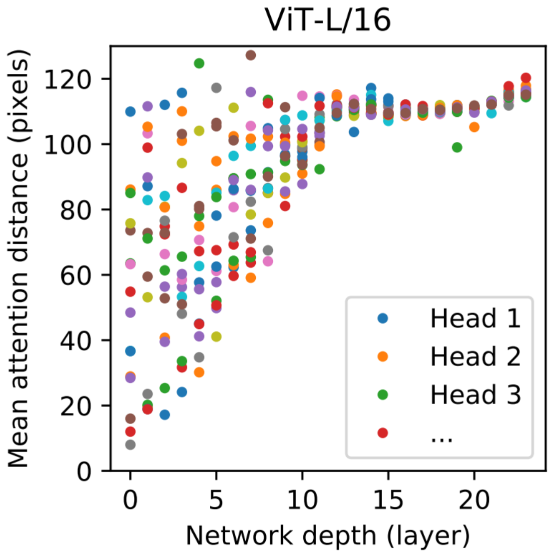


Similar to **CNN filters**





Position embedding includes incorporated **2D structure**

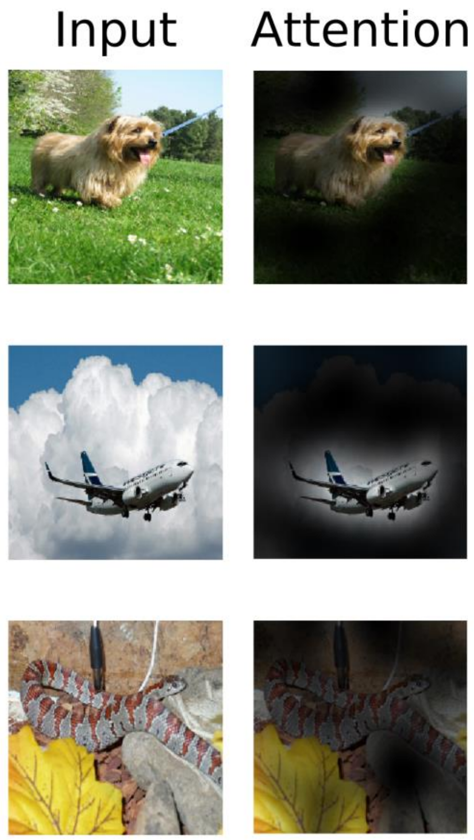


Starts at small scale and step by step increases scope (like CNN)

https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

https://arxiv.org/abs/2010.11929

Vision Transformer Insight



Global self-attention,
in contrast to local
filters of CNN

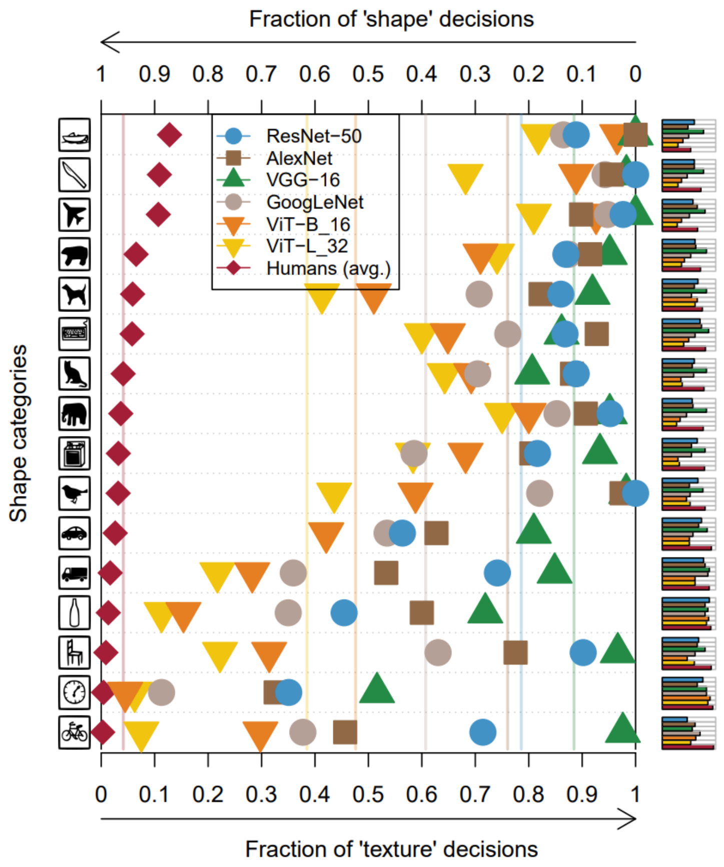
https://arxiv.org/abs/2010.11929

Shape = Cat, Texture = Elephant

Are Vision Transformers **more similar to human vision** than CNNs?

Ongoing research…

Prediction based on shape or on texture?



https://arxiv.org/abs/2105.07197

ImageNet Leaderboard

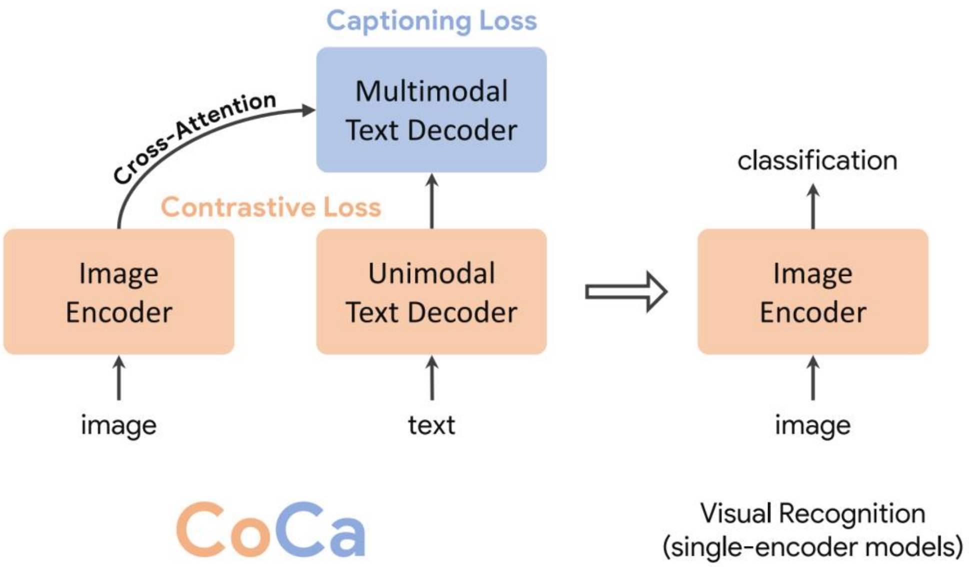
Image Classification on ImageNet:

Rank	Model	Top 1 Accuracy	Top 5 Accuracy	Number of params	Year	Tags
1	CoCa (finetuned)	91.0%		2100M	2022	ALIGN, Transformer, JFT-3B
2	Model soups (BASIC-L)	90.98%		2440M	2022	Conv+Transformer, JFT-3B, ALIGN
3	Model soups (ViT-G/14)	90.94%		1843M	2022	JFT-3B, Transformer
4	ViT-e	90.9%		3900M	2022	Transformer, JFT-3B

https://paperswithcode.com/sota/image-classification-on-imagenet

CoCa

"**Co**ntrastive **Ca**ptioners are Image-Text Foundation Models"



"These results suggest the proposed framework **efficiently combines text training signals** and thus is able to **learn high-quality visual representation** better than the classical single-encoder approach."

https://arxiv.org/abs/2205.01917

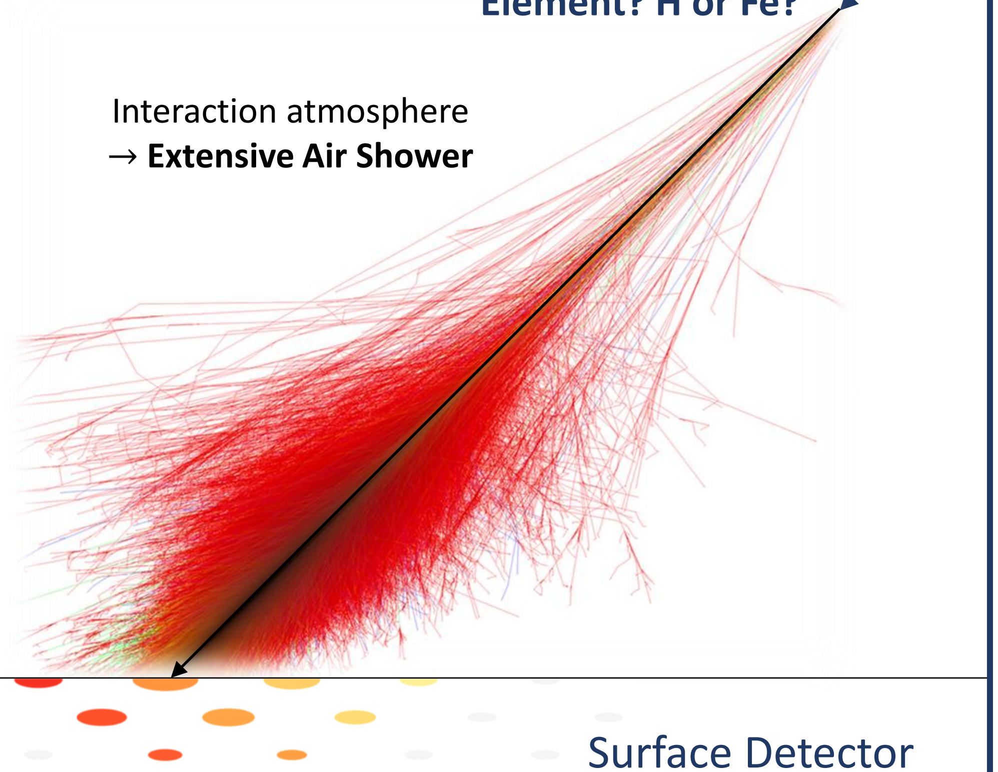
Transformers beat CNNs as state of the art in image recognition!

Application Example: Cosmic-Ray Element Reconstruction at Pierre Auger Observatory

Extensive Air Showers

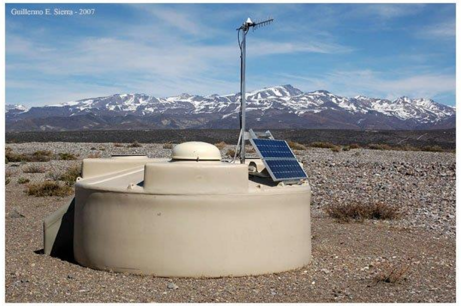
“Primary particle”
Element? H or Fe?

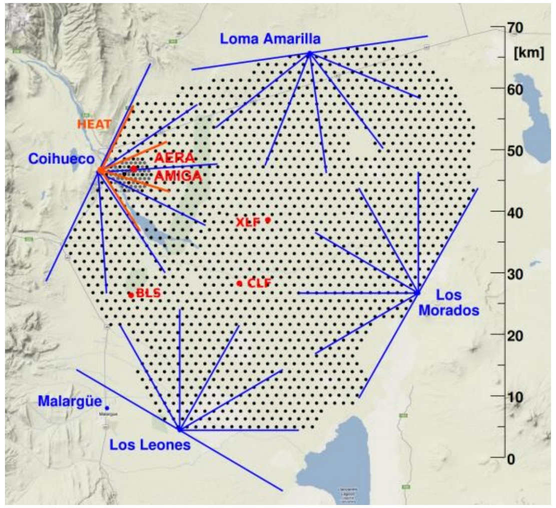
Interaction atmosphere
→ **Extensive Air Shower**



Surface Detector

Pierre Auger Observatory Surface Detector

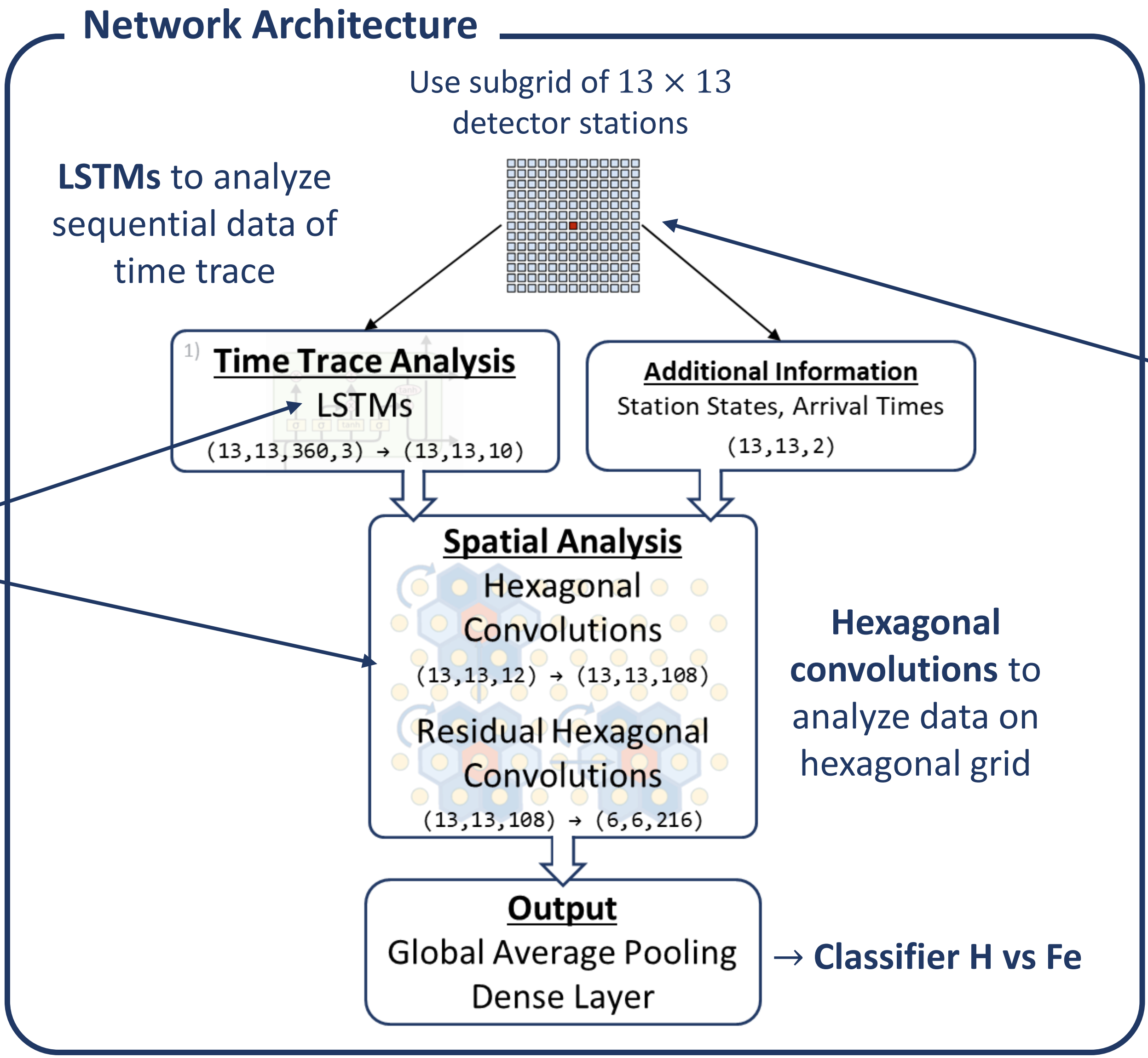




- Located in the *Pampa Amarilla* near Malargüe, Argentina
- Covers an area of roughly $3000\ \mathrm{km}^2$
- **Hexagonal grid** of 1660 water-Cherenkov stations
- Sample air shower footprint

Shower Image Credit: https://www-zeuthen.desy.de/~jknapp/fs/proton-showers.html

Cosmic-Ray Mass Reconstruction Neural Network



Replace with transformer

Signal over time:

Time Traces

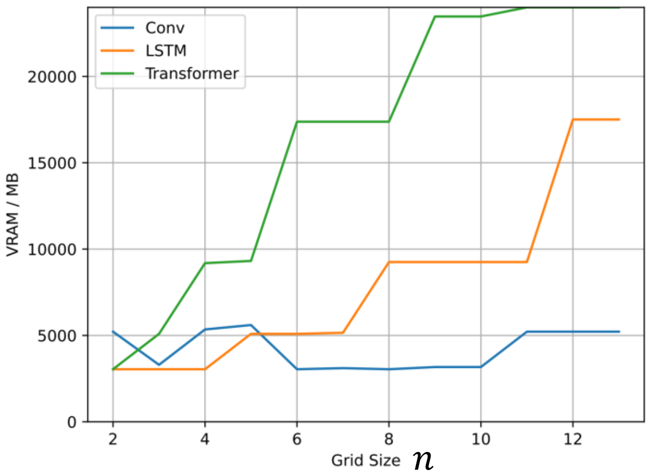
Shower Footprint

1) Image Credit: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

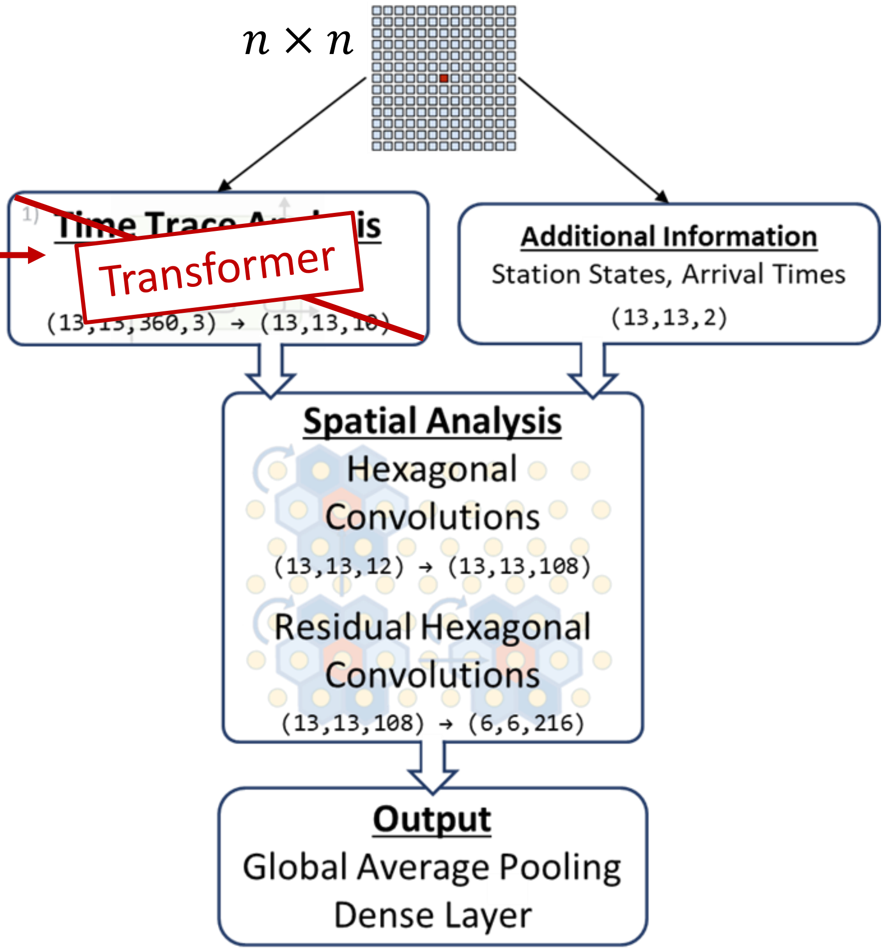
Challenges of Transformer Application

Dimensionality Problem

Attention shape: $(\text{batchsize} \cdot n^2 \cdot N_{\text{head}} \times \underbrace{360 \times 360}_{\mathcal{O}(L^2)})$



Huge VRAM needed!
→ Not practical



Exemplary alternatives to reduce $\mathcal{O}(L^2)$:

Reformer: $\mathcal{O}(L \cdot \log L)$
https://arxiv.org/abs/2001.04451

Sparse Transformers: $\mathcal{O}(L \cdot \sqrt{L})$
https://arxiv.org/abs/1904.10509

→ Challenging to implement

Perceiver: $\mathcal{O}(kL)$
https://arxiv.org/abs/2107.14795

→ **Very easy to implement** (see hands-on session)

many more …
See https://arxiv.org/abs/2009.06732 and https://arxiv.org/abs/2011.04006

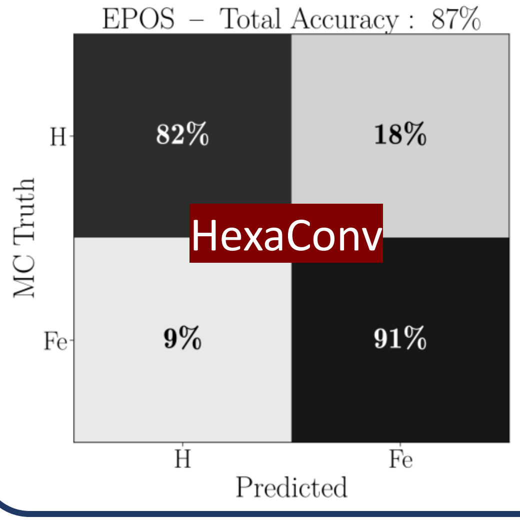
⇒ **Default attention quickly needs very large amounts of VRAM** if data structure is too large, **alternatives exist**

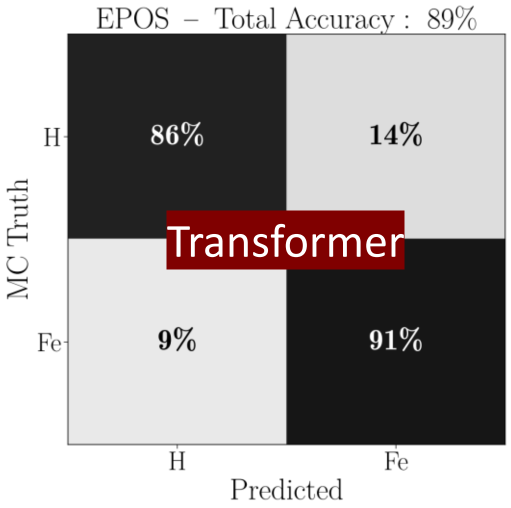
1) Image Credit: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

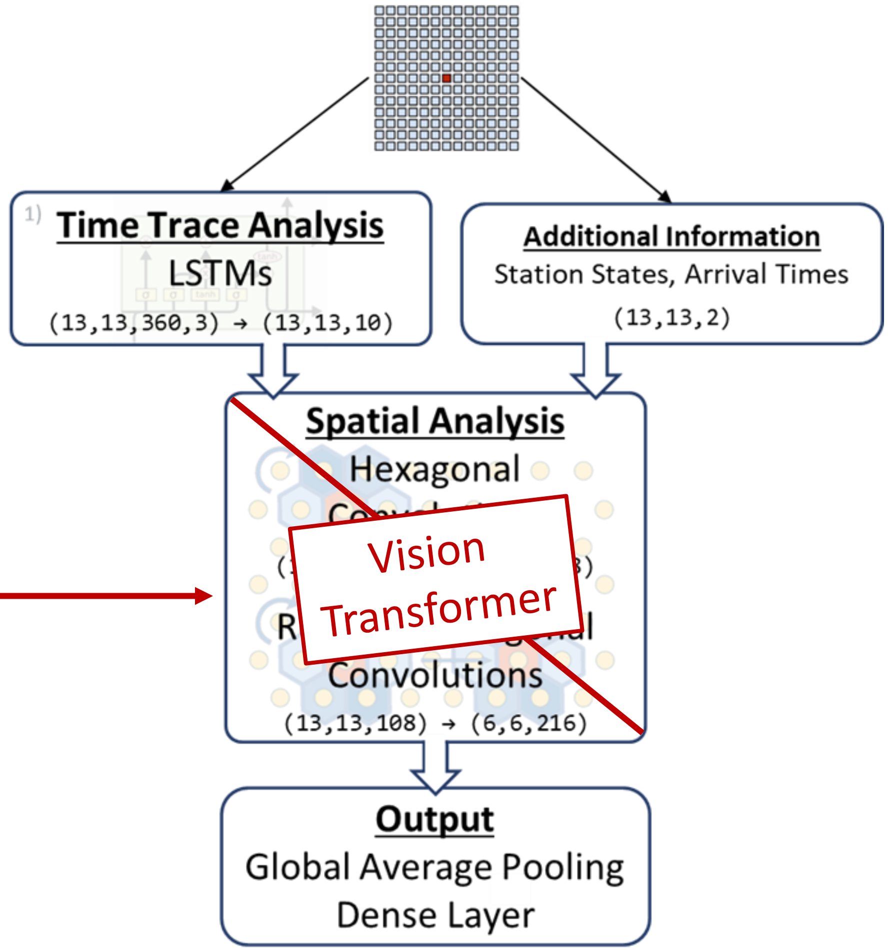
Chances of Transformer Application

Spatial Transformer

- Use simple, **out of the box vision transformer** to replace hexagonal convolution
- **Loses inherent knowledge of**:
  - Hexagonal symmetry
  - 2D structure
- **Increases number of parameters** by factor 17
- **Reduces training time by 50%**
- **Increases performance!**





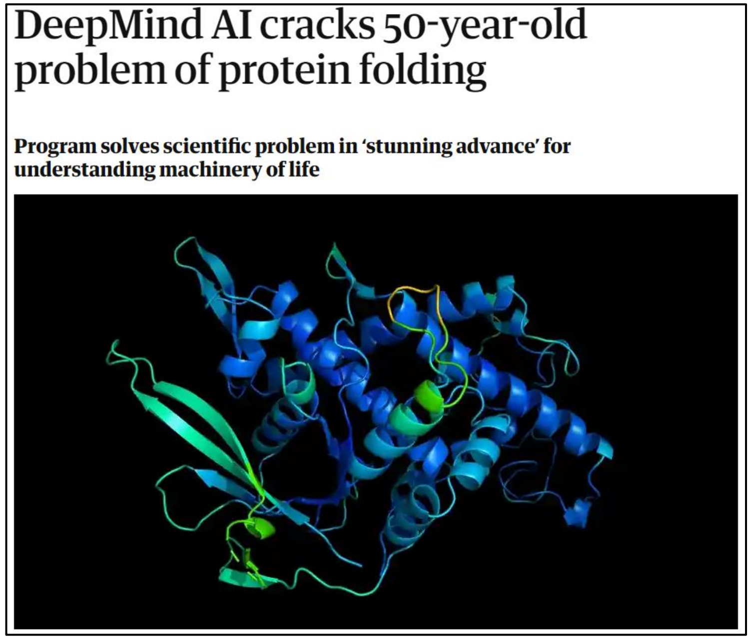


⇒ **Domain agnostic transformer** can be trained quickly, and **outperforms approach focused on data symmetry** (similar to Vision Transformer on images)

1) Image Credit: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

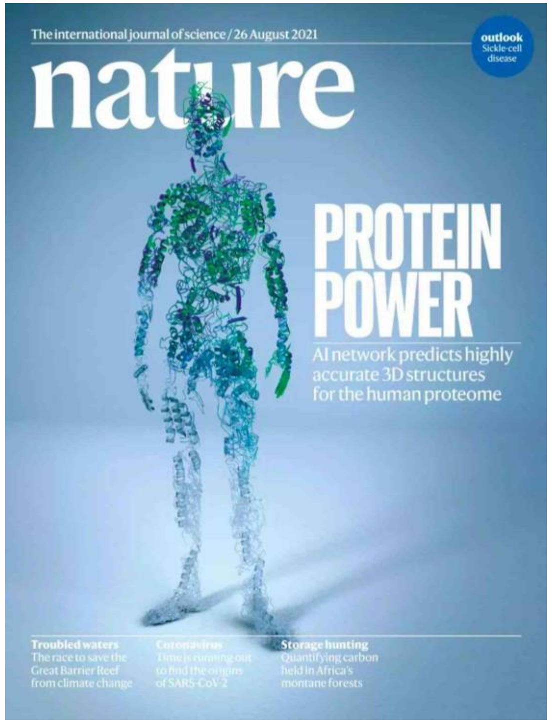
Transformers in Science

Similar to other fields, transformers are used **instead of LSTMs or CNNs**, as well as for **new applications:**



DeepMind AI cracks 50-year-old problem of protein folding

Program solves scientific problem in 'stunning advance' for understanding machinery of life



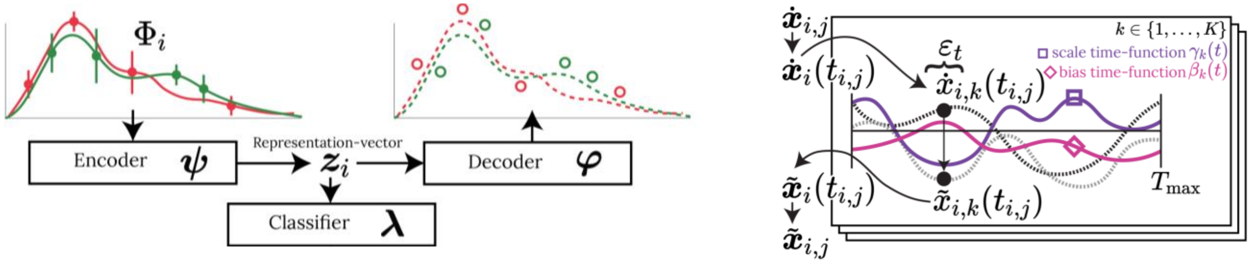
Famous science application:
Protein unfolding

Applications in Astro- and Particle Physics are becoming more common:
(The following list contains some of the results from arXiv and is not extensive)

Time Sequence Analyses

https://arxiv.org/pdf/2201.08482.pdf

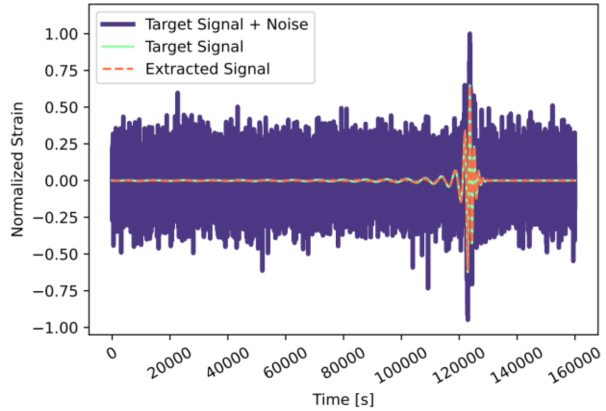
Deep Attention-Based Supernovae Classification of Multi-Band Light-Curves



TimeFiLM

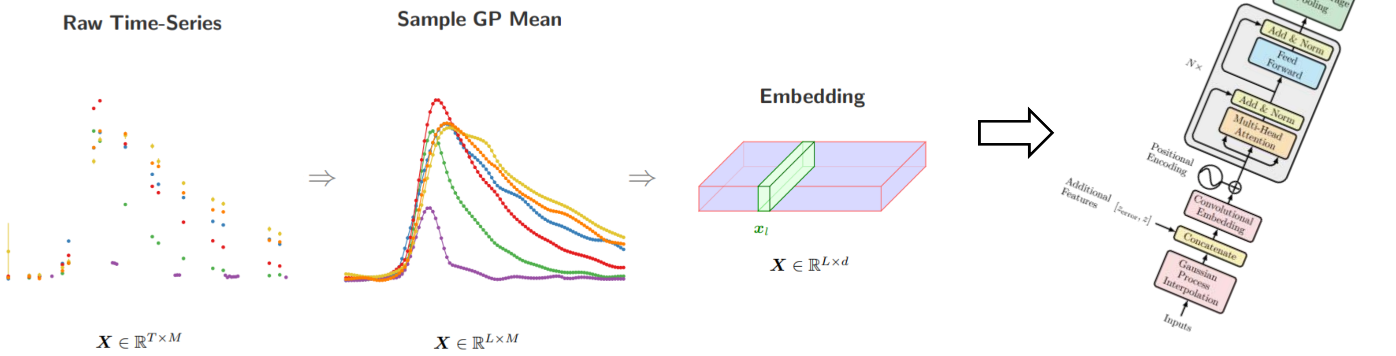
https://arxiv.org/pdf/2207.07414.pdf

SPACE-BASED GRAVITATIONAL WAVE SIGNAL DETECTION AND EXTRACTION WITH DEEP NEURAL NETWORK



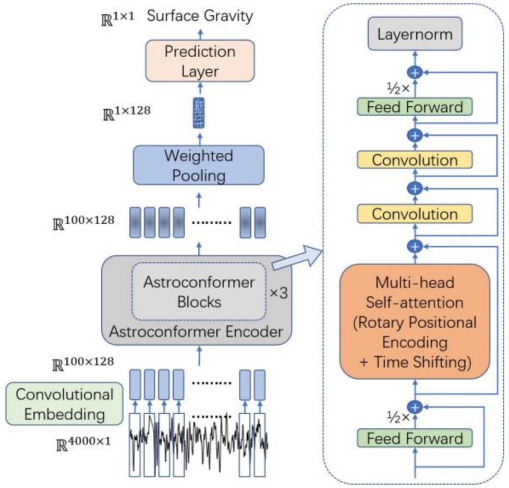
https://arxiv.org/pdf/2105.06178.pdf

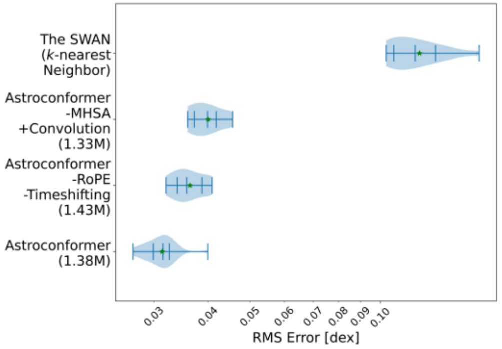
Paying Attention to Astronomical Transients: Photometric Classification with the Time-Series Transformer



Time Sequence Analyses

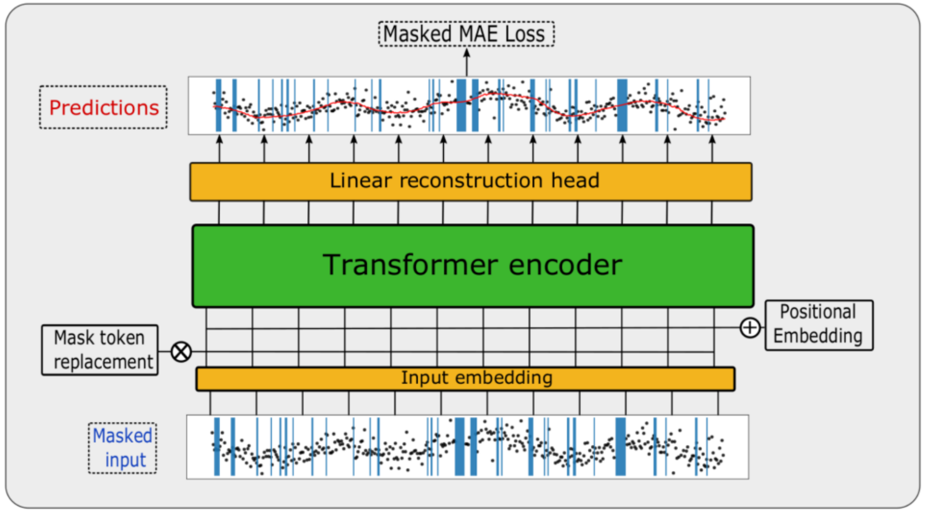
Astroconformer: Inferring Surface Gravity of Stars from Stellar Light Curves with Transformer



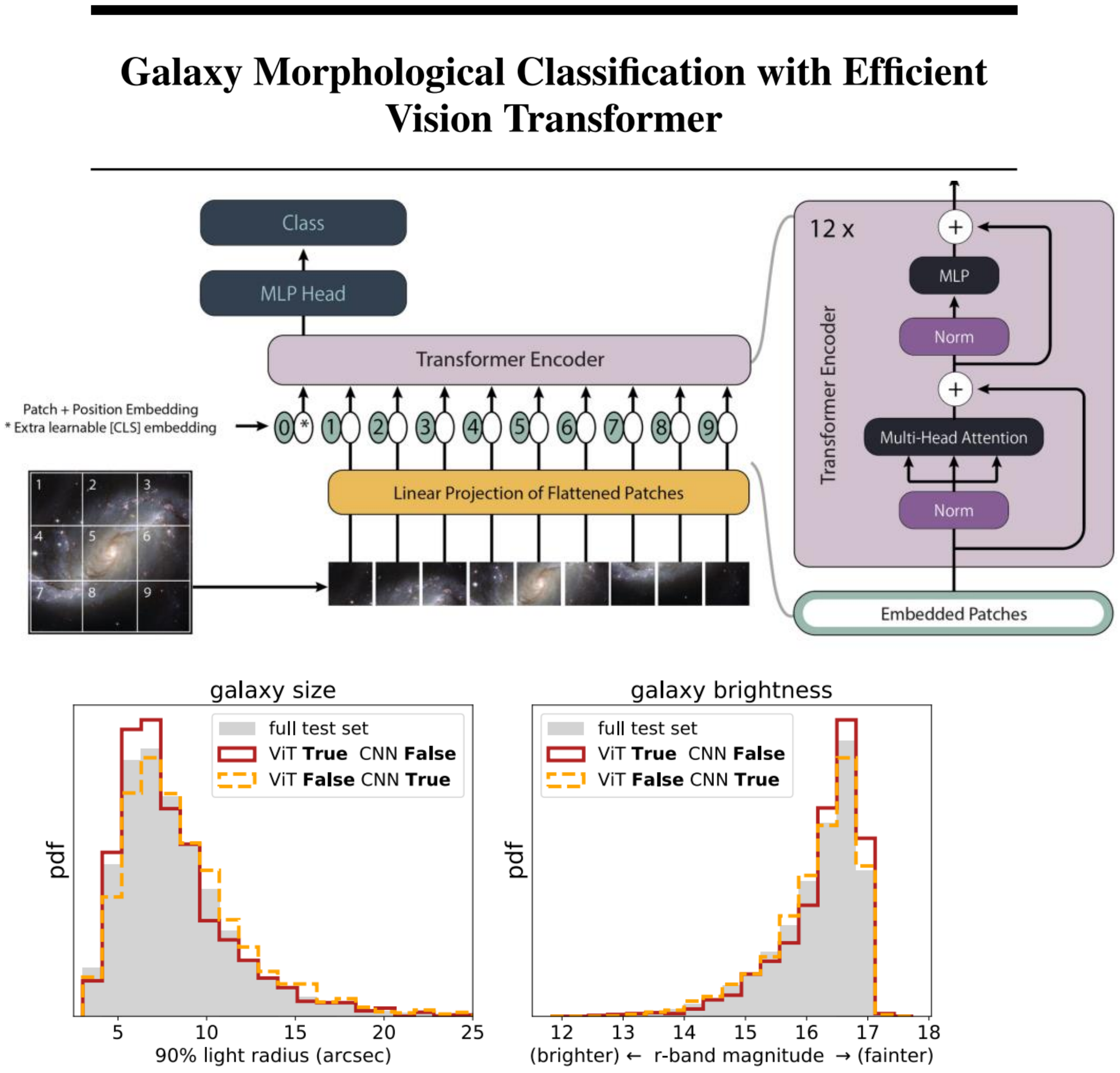


https://arxiv.org/pdf/2207.02777.pdf

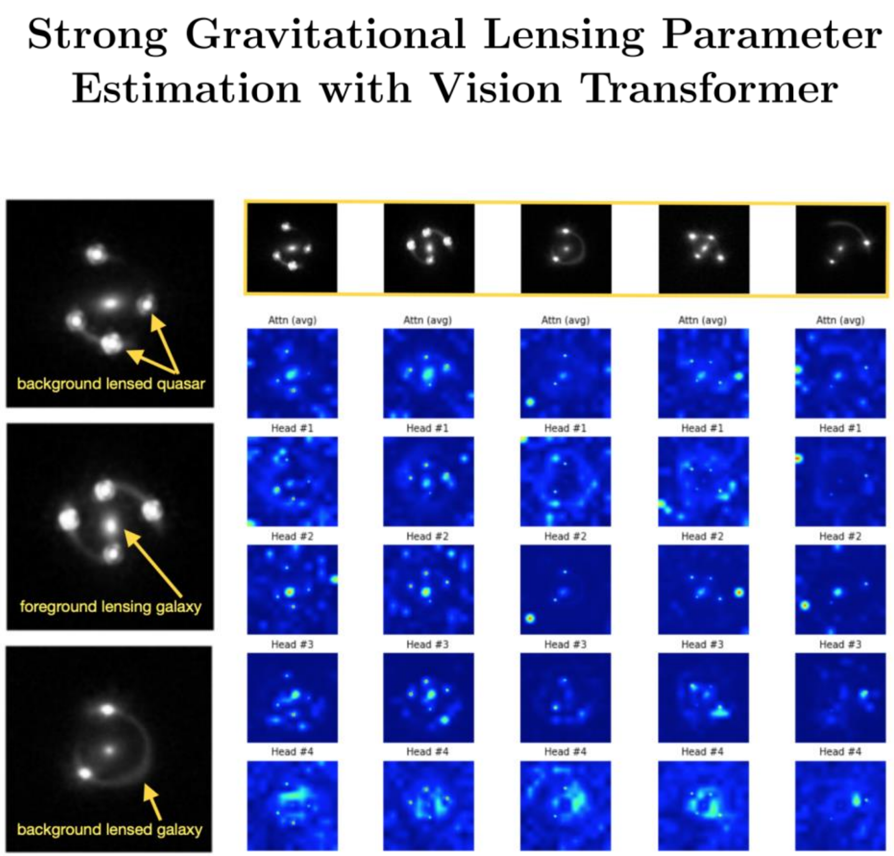
Don't Pay Attention to the Noise: Learning Self-supervised Representations of Light Curves with a Denoising Time Series Transformer



Vision Transformer Applications



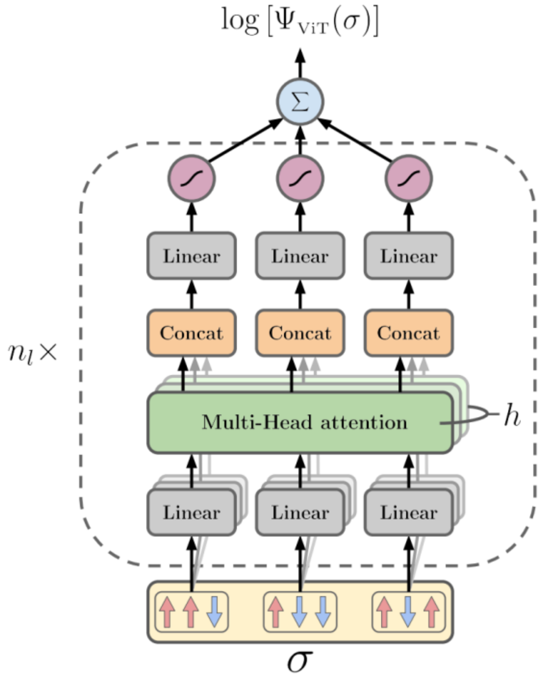
https://arxiv.org/pdf/2110.01024.pdf

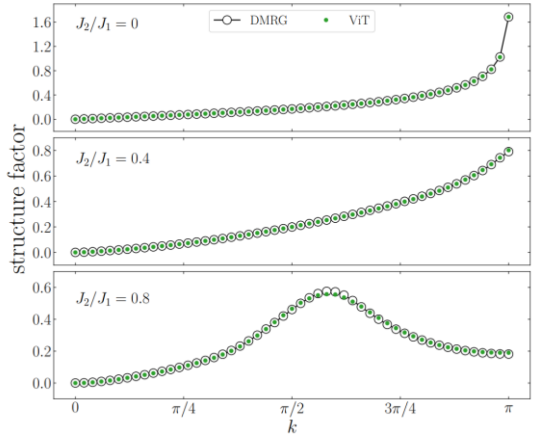


https://arxiv.org/pdf/2210.04143.pdf

Other Applications

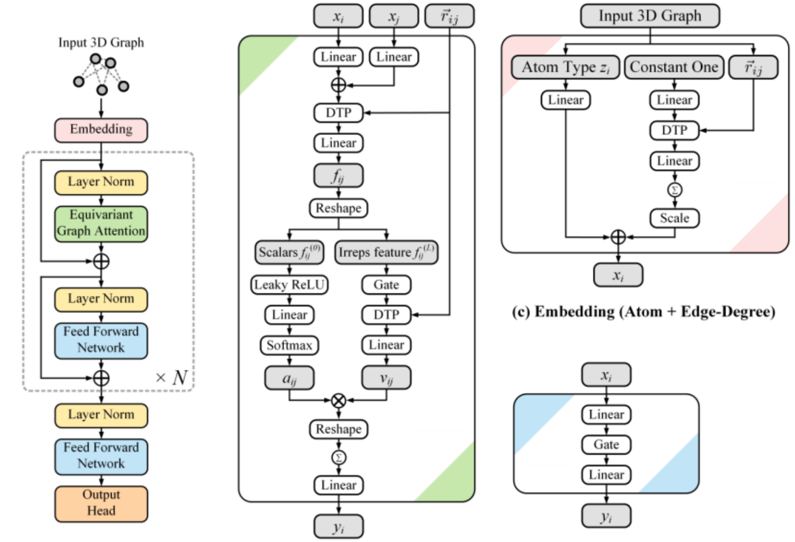
Transformer variational wave functions for frustrated quantum spin systems





https://arxiv.org/pdf/2211.05504.pdf

Equiformer: Equivariant Graph Attention Transformer for 3D Atomistic Graphs



(a) Equiformer Architecture (b) Equivariant Graph Attention (c) Embedding (Atom + Edge-Degree) (d) Feed Forward Network

Fine-tuning Vision Transformers for the Prediction of State Variables in Ising Models

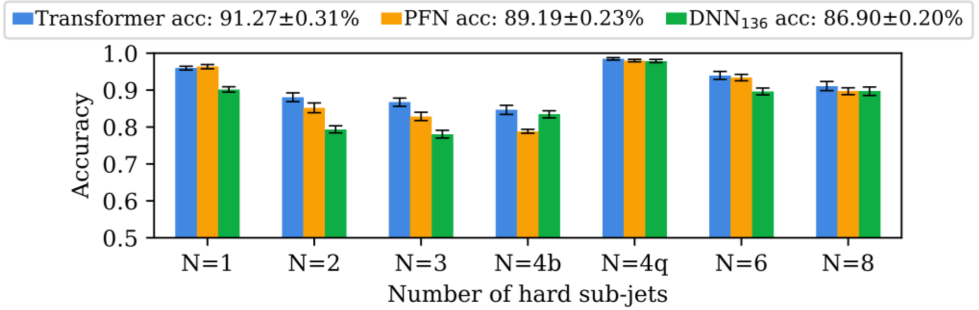
https://arxiv.org/pdf/2109.13925.pdf

System/Boundary Conditions	ViT	ResNet-18	ResNet-50
Periodic (Ferromagnetic)	**0.934** ± **.008**	0.865 ± .034	0.907 ± .021
Periodic (Anti-ferromagnetic)	**0.935** ± **.012**	0.906 ± .016	0.899 ± .026
Skewed (Ferromagnetic)	**0.931** ± **.021**	.886 ± .019	0.917 ± .009
Anti-Periodic (Ferromagnetic)	**0.921** ± **.013**	0.91 ± .021	0.917 ± .008

Particle Physics

https://arxiv.org/pdf/2202.00723.pdf

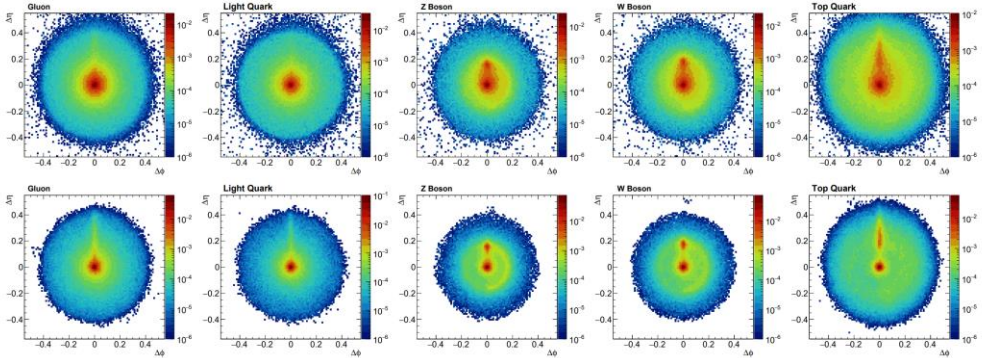
Resolving Extreme Jet Substructure



https://arxiv.org/pdf/2102.05073.pdf

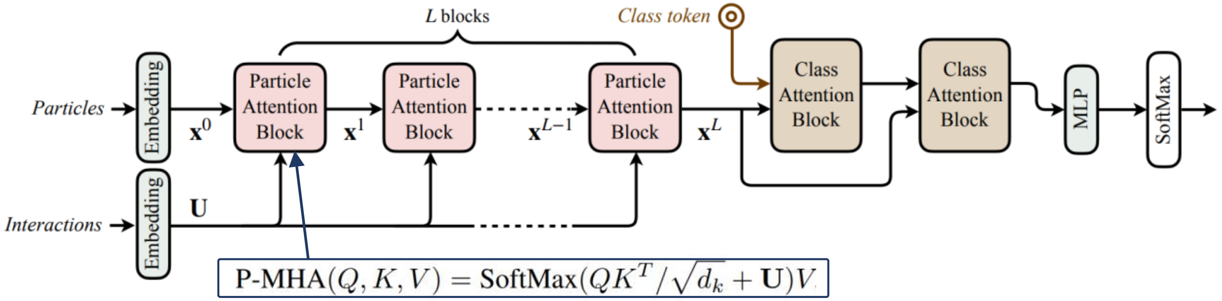
Point Cloud Transformers applied to Collider Physics

Attention layers



https://arxiv.org/pdf/2202.03772.pdf

Particle Transformer for Jet Tagging



JetClass Classification

	Accuracy	# params	FLOPs
PFN	0.772	86.1 k	4.62 M
P-CNN	0.809	354 k	15.5 M
ParticleNet	0.844	370 k	540 M
ParT	**0.861**	2.14 M	340 M
ParT (plain)	0.849	2.13 M	260 M

Top-Tagging

	Accuracy
P-CNN	0.930
PFN	—
ParticleNet	0.940
JEDI-net (w/ $\sum O$)	0.930
PCT	0.940
LGN	0.929
rPCN	—
LorentzNet	0.942
ParT	0.940
ParticleNet-f.t.	0.942
ParT-f.t.	**0.944**

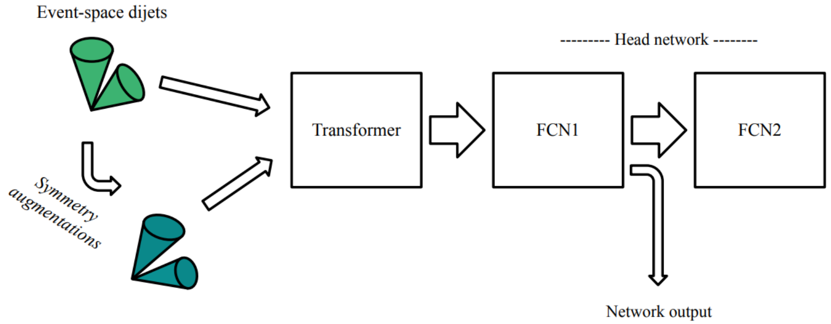
(Also best in quark-gluon tagging)

"P-MHA, an augmented version that can also exploit the pairwise particle interactions directly […]
*Essentially, we add the interaction matrix **U** […]. This allows P-MHA to incorporate particle interaction features designed from physics principles …"*

Particle Physics

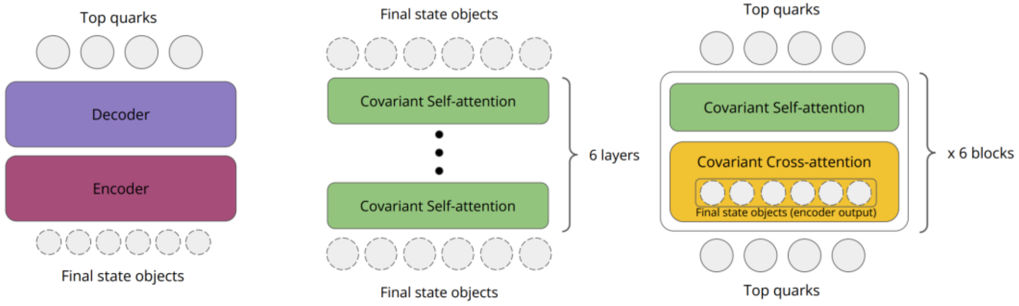
https://arxiv.org/pdf/2205.10380.pdf

Self-supervised Anomaly Detection for New Physics



https://arxiv.org/pdf/2203.05687.pdf

A Holistic Approach to Predicting Top Quark Kinematic Properties with the Covariant Particle Transformer



(a) CPT (b) Encoder (c) Decoder

Contrastive Learning

https://arxiv.org/pdf/2108.04253.pdf

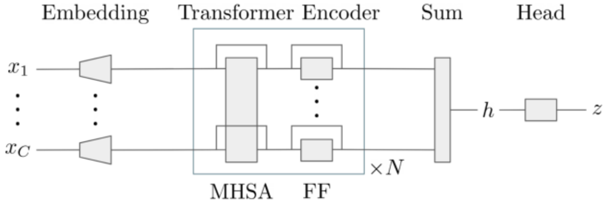
Symmetries, Safety, and Self-Supervision

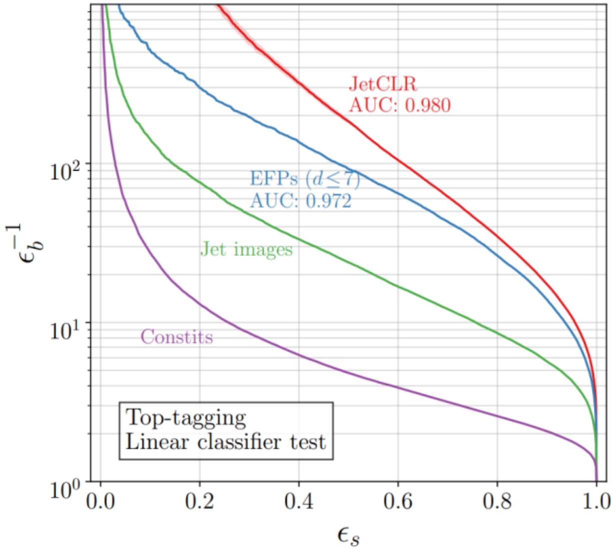
The goal of our network is to define a mapping between the jet constituents and a representation space,

$$f : \mathcal{J} \to \mathcal{R} , \tag{1}$$

which is, both,

1. invariant to symmetries and theory-driven augmentations, and
2. discriminative within the dataset it is optimized on.





Summary

- **Transformer**: Based on **attention mechanism**
- **New State-of-the art** in many fields
  - Natural language processing
  - Image recognition
  - …
- **Large amounts of VRAM** needed to analyze **long sequences**
  - **Alternatives (Reformer, Sparse Transformer, Perceiver)** exist, but some are less accessible
- Easily applicable to many different tasks, due to being **domain agnostic**
  - **Quickly spreading** to more branches of machine learning
  - Usage of transformers in **physics** at an early stage with **promising results**