The sequential architecture & AI for scientific computing

Haoyi Zhou

Thursday, Feb 27, 2025

Beihang University

- Content -

- **Sequential Modeling: from Science**
- The Transformer models
- The modern architecture
- Some applications on scientific sequences
- Future direction

Tycho → Kepler → Newton

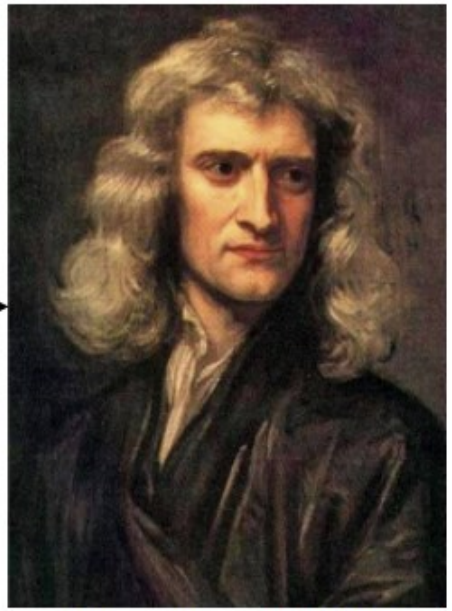
Three stages of transformation including data collection, model learning and model interpretation, which align closely with the development trajectory of AI.

Tycho Brahe (1546-1610)

Observation
Data Collection

Johannes Kepler (1571-1630)

Analyzation
Model Learning



Isaac Newton (1643-1727)

Explanation
Model Interpretation

https://arxiv.org/pdf/2111.12210.pdf

Outlook on Scientific Research Paradigm in 2050

US Department of Energy National Laboratory Advanced Scientific Computing Research Group Reported in 2022, looking forward to the future form of scientific research in 2050

The Theory Machine Transforms Science

Intelligently complete experimental data analysis, theoretical derivation, and simulation calculations

Delphi: The universal knowledge map

Intelligently summarize and extract scientific literature and experimental reports, help scientists analyze and predict research directions, and propose new scientific hypotheses

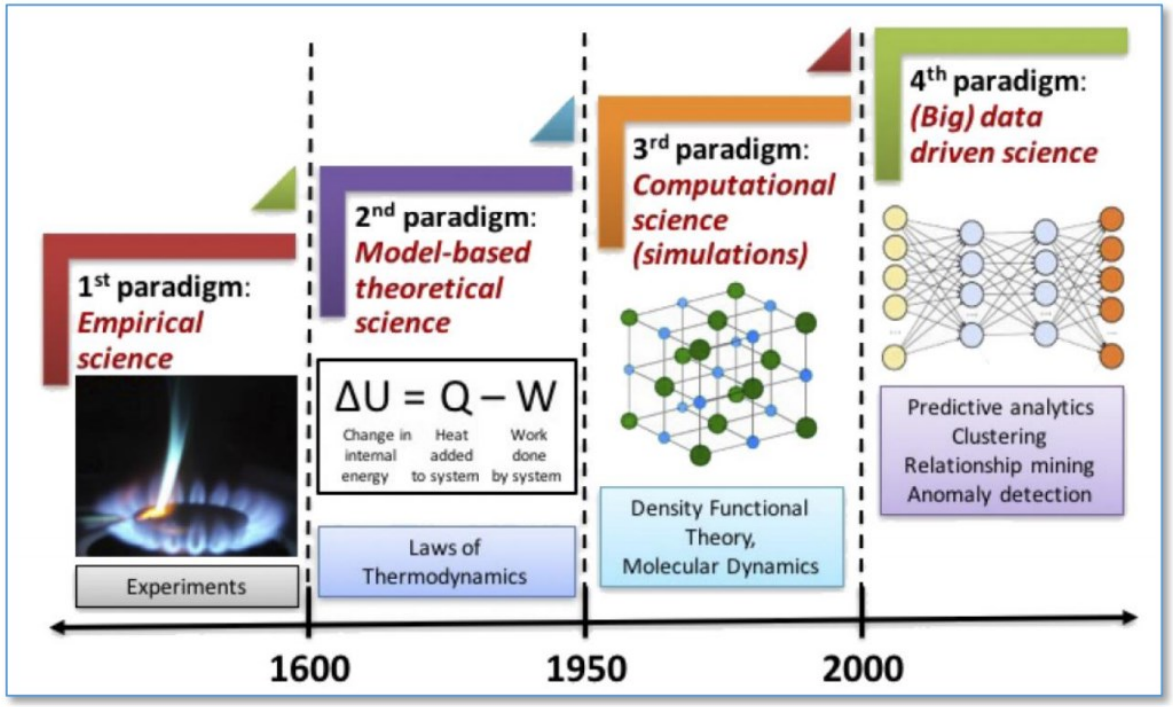
Smart self-driving facilities automate experimentation

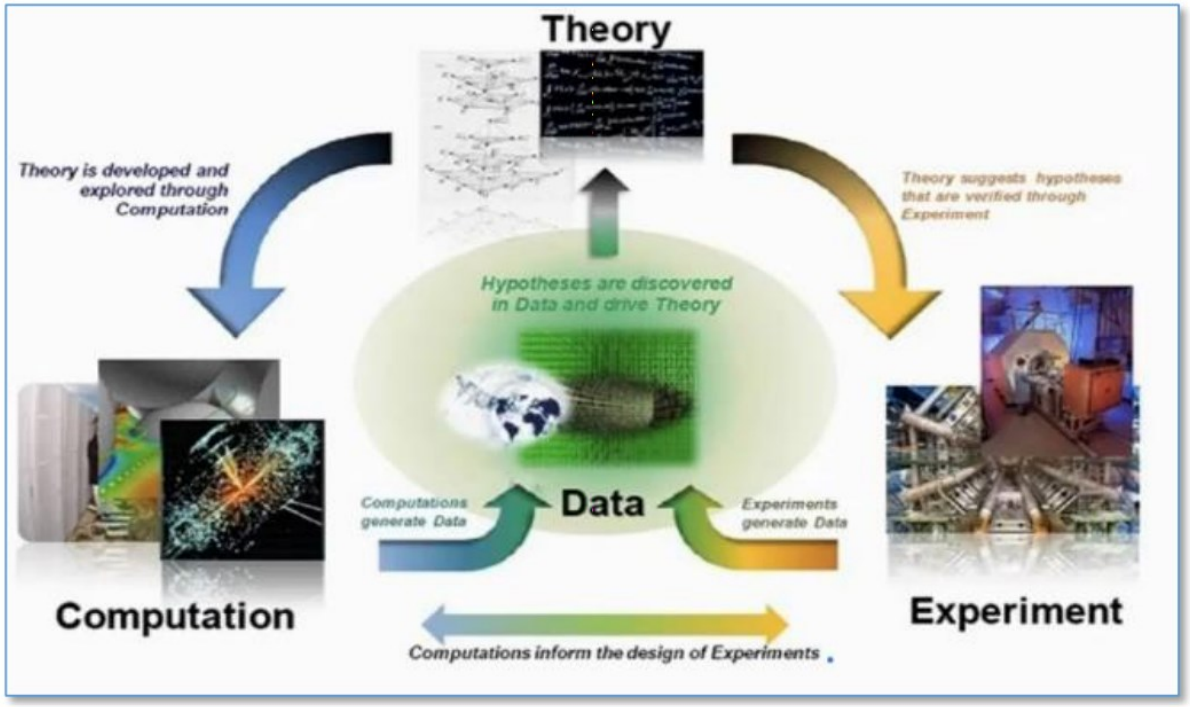
Like autonomous driving, independently completing scientific experiments and becoming an intelligent assistant for experimental scientists

Universal Data Service Make All Data Easily Accessible

Linked Facilities Form a Discovery Cloud

AI is becoming a New Tool for Scientific Research



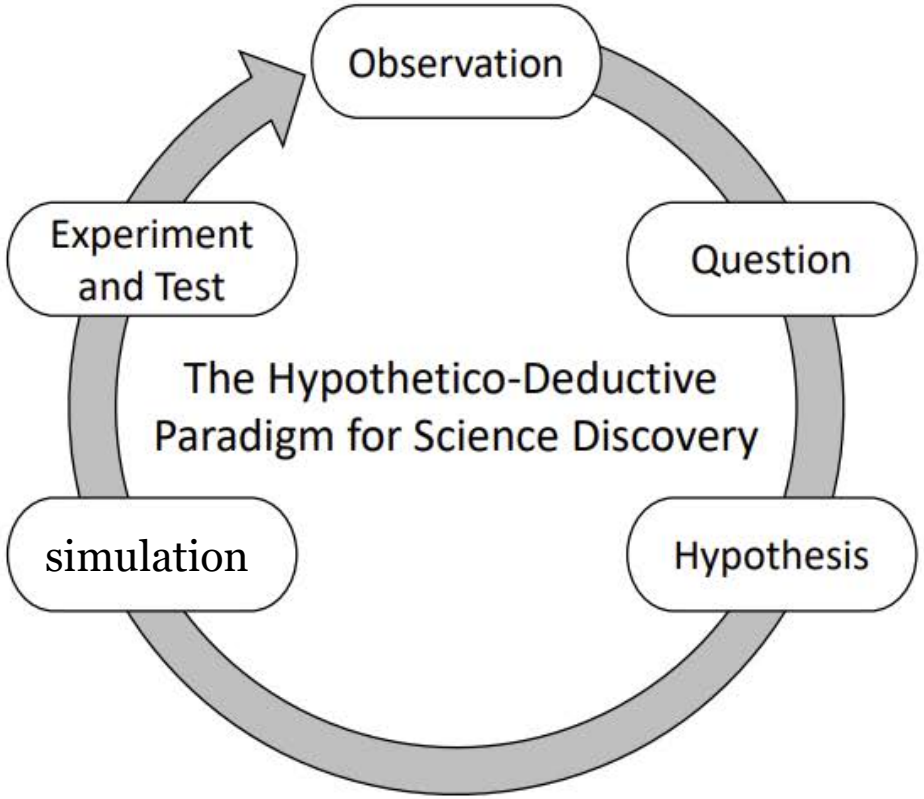


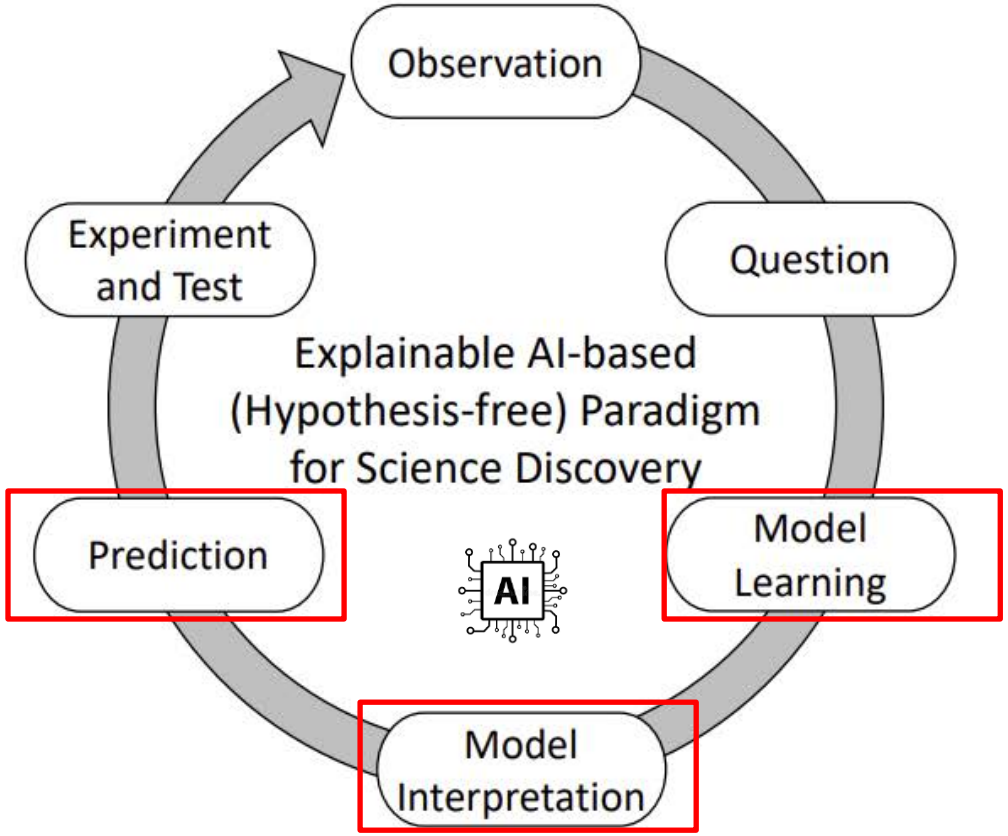
Experimental Science → Theoretical Science → Computing Science →

AI pushes scientific research into **the paradigm 4**

Hypothesis-free paradigm for Science Discovery

The hypothesis and simulation stage are replaced by AI models, which can learn underlying pattern in the observed data and directly output possible physical interpretation and prediction.



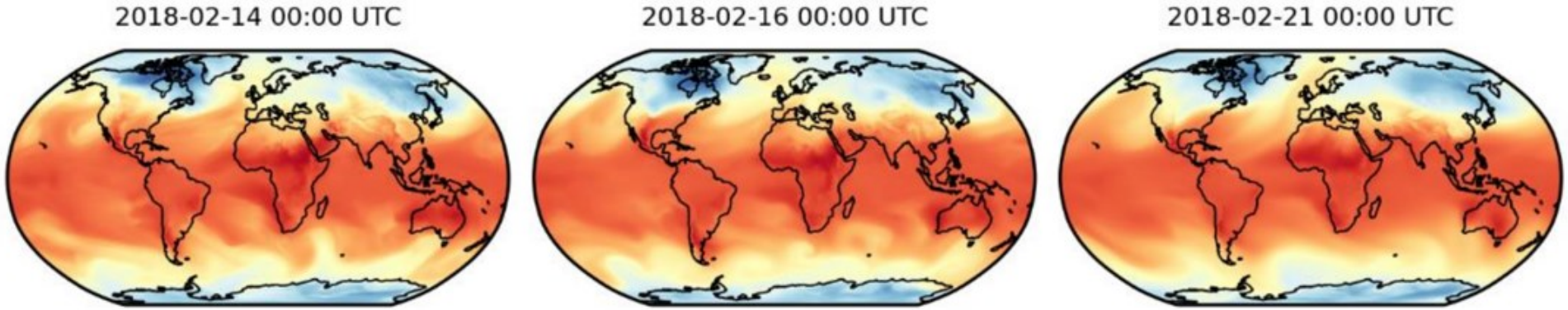


https://arxiv.org/pdf/2111.12210.pdf

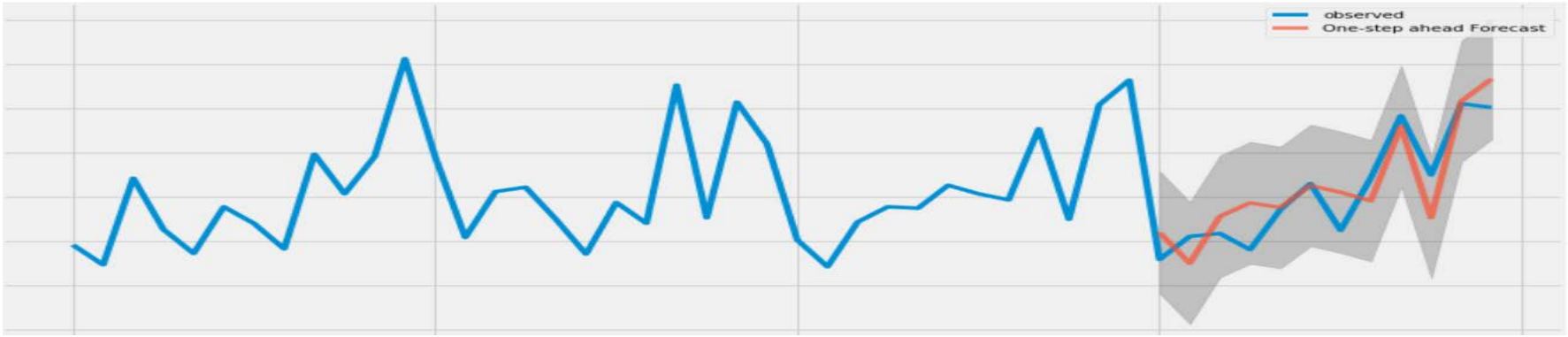
Time-series and dynamic system

Physical dynamical systems are essentially time series controlled by explicit equations

Dynamic system



Time-series



Sequential Data: The Requirement for Computation

Intelligent computing of sequential data is a crucial for the support of industrial development.

A320: 12,000 parameters, A350: nearly 600,000 parameters.
Boeing 787: One flight generates 0.5TB of data (including only cabin pressure, etc.).

Social Activities

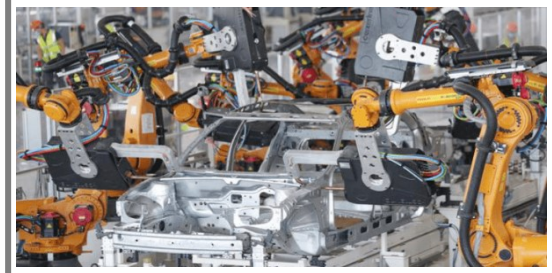
Behavioral data

Smart cities

Complex sensor data

Intelligent Manufacturing

Aerospace



Robotic automation

Continuous high-frequency data

Infrastructure

Power and energy industry

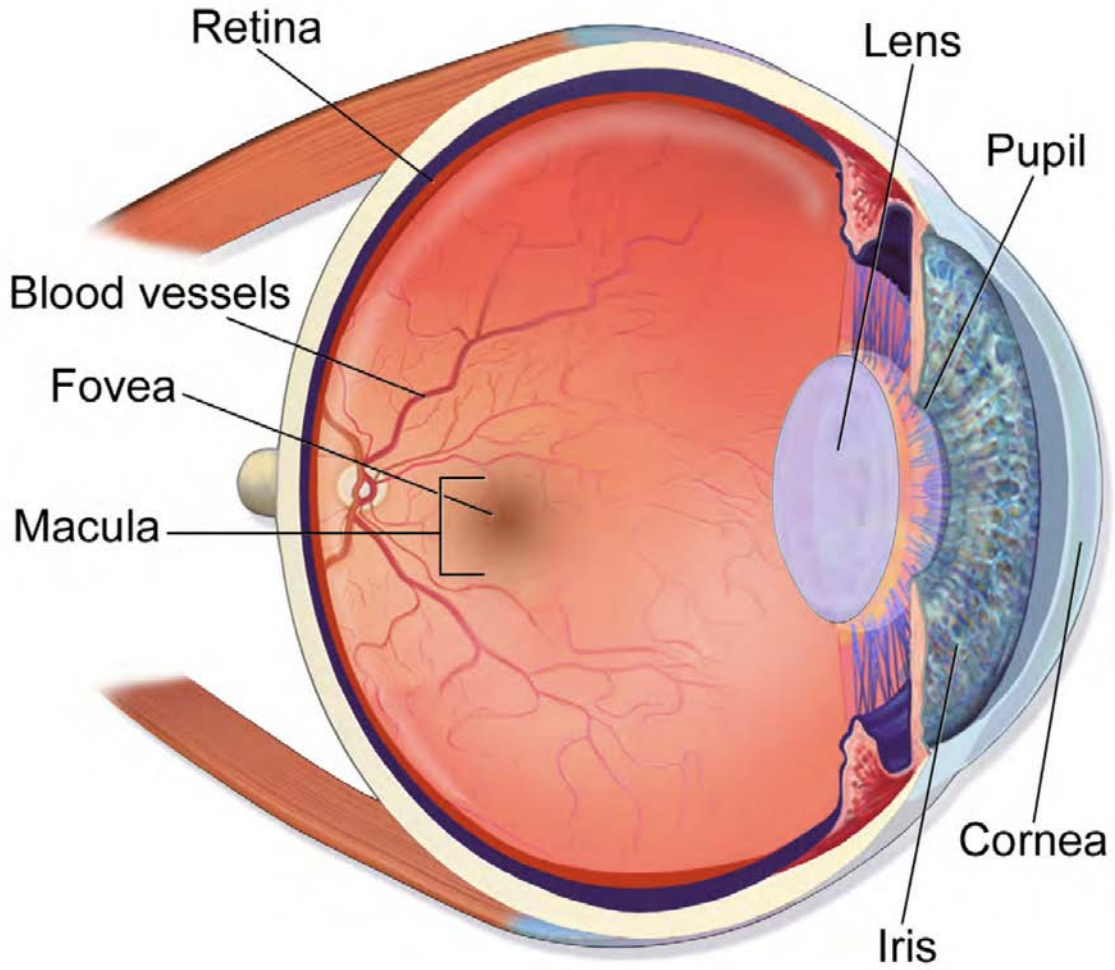
UAV inspection

Working condition data

- Content -

- Sequential Modeling: from Science
- **The Transformer Model**
- The modern architecture
- Some applications on scientific sequences
- Future direction

What is the attention?



* Images are from google search engine.

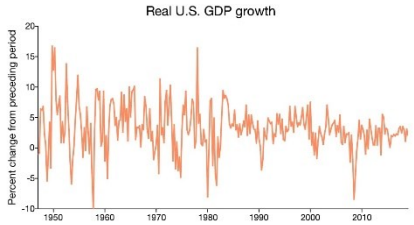
How do we tackle the forecasting problem?

Problem Setting: LSTF predicts the long sequence output from massive long sequence inputs.

How do we plan gifts for Christmas day?

We will have some feelings of what you have seen when it comes to the prediction.

Attention Mechanism is arguably one of the most powerful concepts in the deep learning field nowadays. It is based on a common intuition that we "attend to" a certain part when processing a large amount of information.



The development of attention models

2015 - 2017 Attention

2017 - Now Transformer

Transformer-xl

Bahdanau attention

- First Proposed 《Neural machine translation by jointly learning to align and translate》

Luong attention

- Propose Several Approaches 《Effective Approaches to Attention-based Neural Machine Translation》

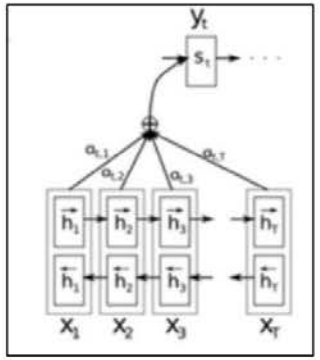
Soft/Hard attention

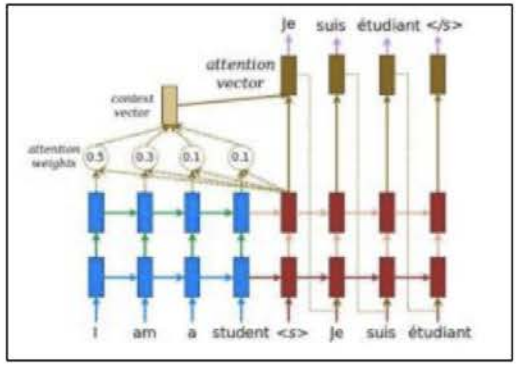
- Apply to Image Caption 《Show, Attend and Tell: Neural Image Caption Generation with Visual Attention》

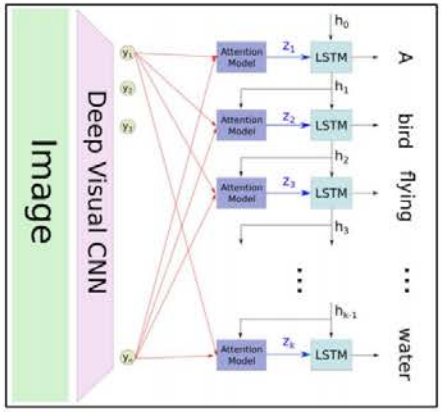
Self-attention Multihead attention

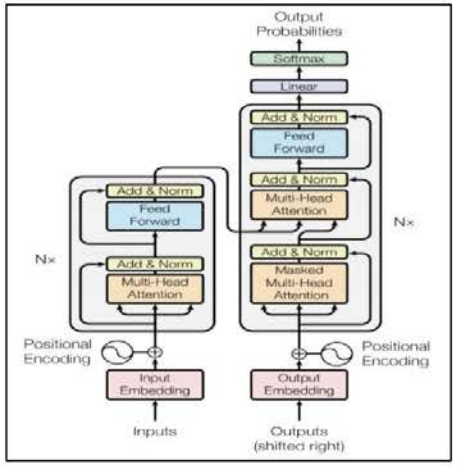
- Propose Transformer 《Attention is all you need》

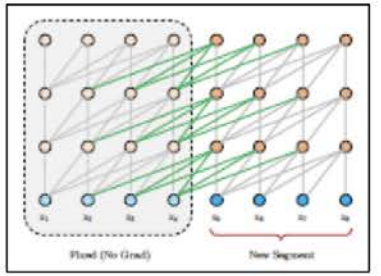
《TRANSFORMER-XL: ATTENTIVE LANGUAGE MODELS BEYOND A FIXED-LENGTH CONTEXT》











Figures are from Aliyun slides.

Take the NLP as an example

https://jalammar.github.io/illustrated-transformer/

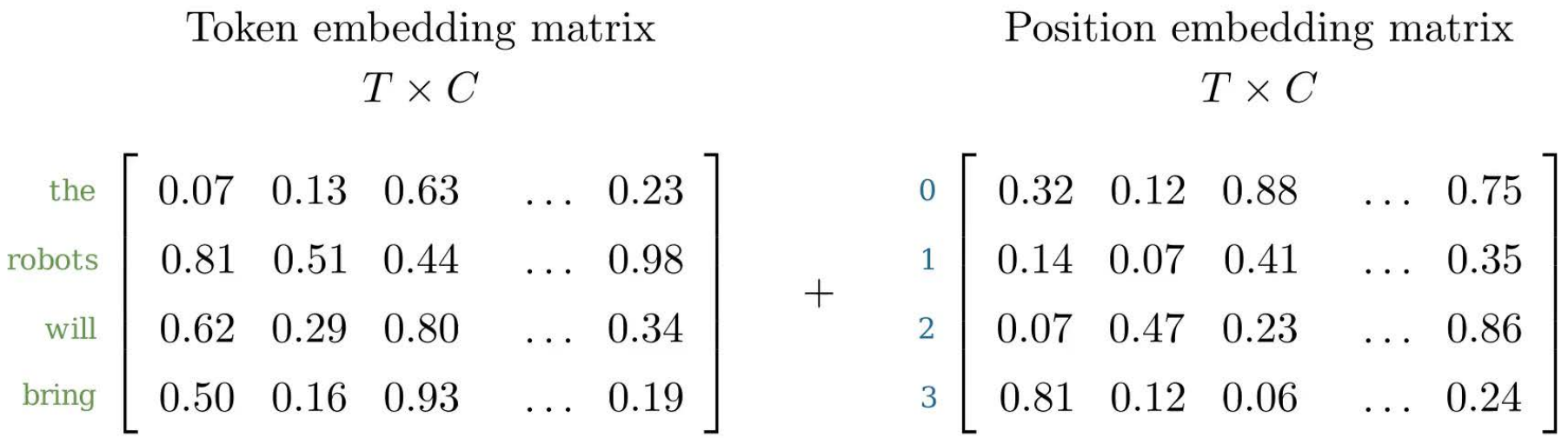
① Embeddings: Numbers more than Words

Token embeddings

① Embeddings: Numbers more than Words

Position embeddings

① Embeddings: Numbers more than Words

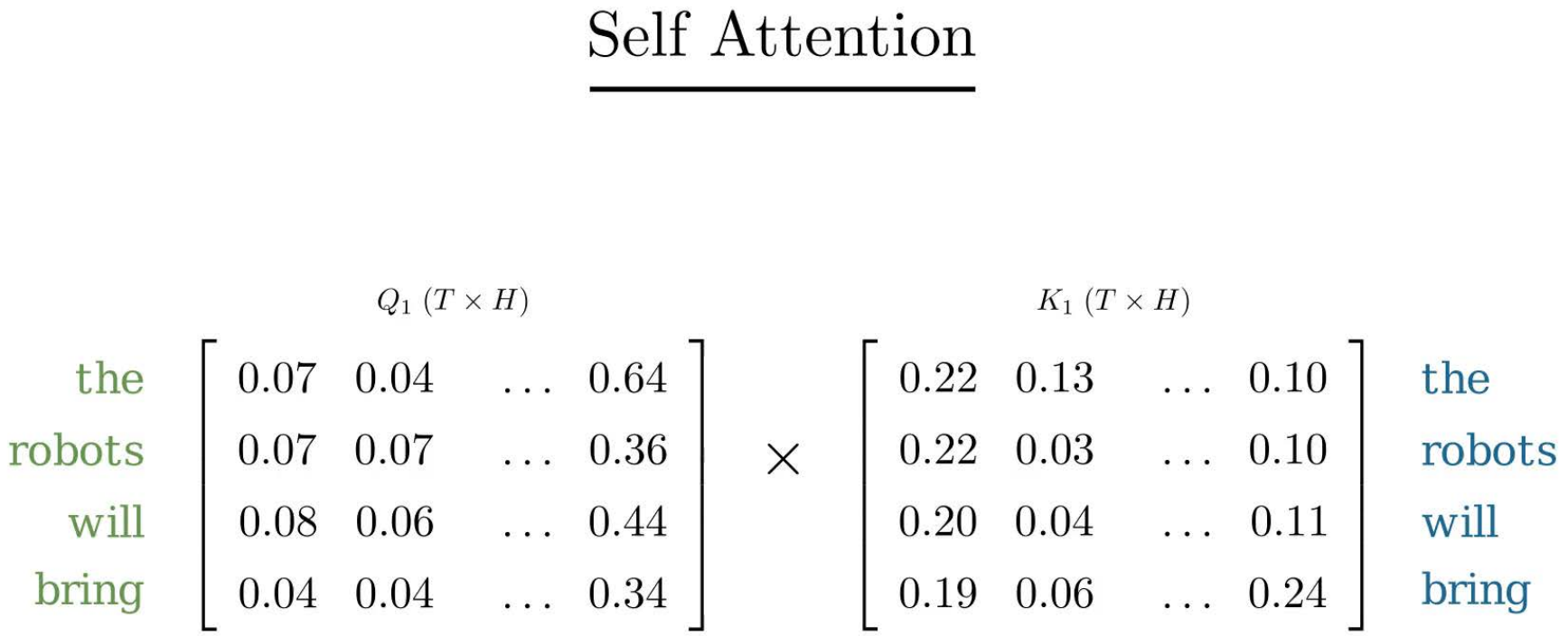


② Queries, keys and values

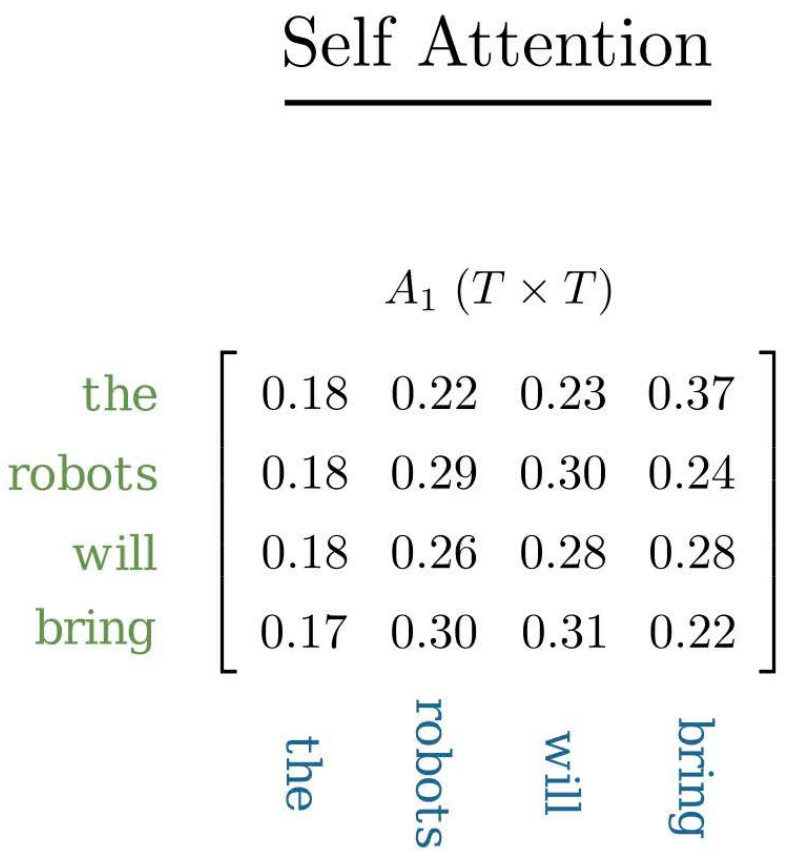
Query, key, value vectors for each word

$$
\begin{array}{r}
 \\
\text{the} \\
\text{robots} \\
\text{will} \\
\text{bring}
\end{array}
\overset{T \times C}{\begin{bmatrix}
0.39 & 0.25 & \dots & 0.98 \\
0.95 & 0.58 & \dots & 1.33 \\
0.69 & 0.77 & \dots & 1.20 \\
1.31 & 0.28 & \dots & 0.43
\end{bmatrix}}
\times
\overset{W_q\ (C \times C)}{\begin{bmatrix}
0.02 & 0.21 & \dots & 0.37 \\
0.03 & 0.02 & \dots & 0.05 \\
0.29 & 0.07 & \dots & 0.99 \\
0.38 & 0.37 & \dots & 0.28 \\
\vdots & \vdots & \vdots & \vdots
\end{bmatrix}}
=
$$

③ “Pay” Attention



④ “Apply” Attention



⑤ Two heads are better than one

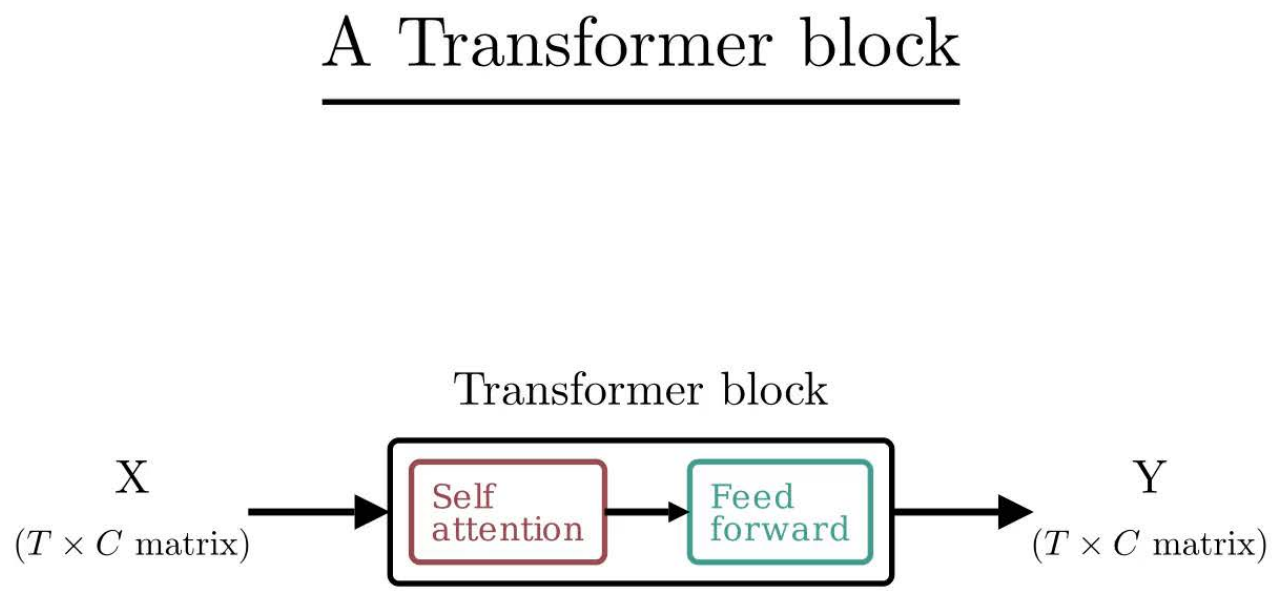
Split Q, K and V matrices into "heads"

$$Q\ (T \times C)$$

$$\begin{bmatrix} 0.07 & 0.04 & \dots & 0.64 \\ 0.07 & 0.07 & \dots & 0.36 \\ 0.08 & 0.06 & \dots & 0.44 \\ 0.04 & 0.04 & \dots & 0.34 \end{bmatrix}$$

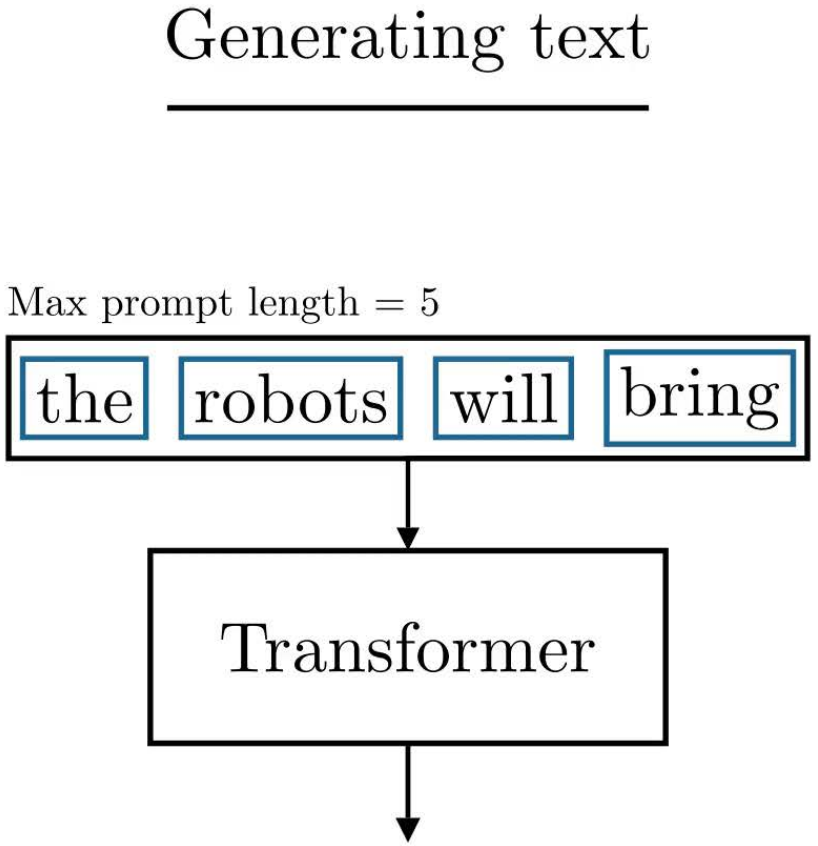
⑥ Putting all heads together

⑦ Stacking the attention block



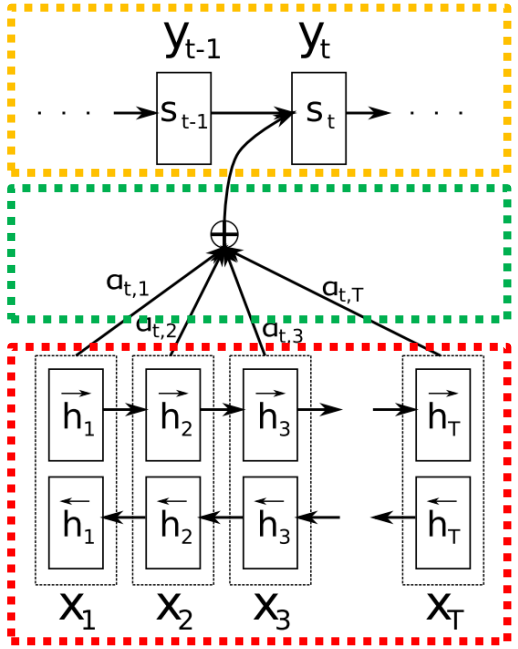
⑧ Making the prediction

⑨ Text Generator & Window



The Transformer model

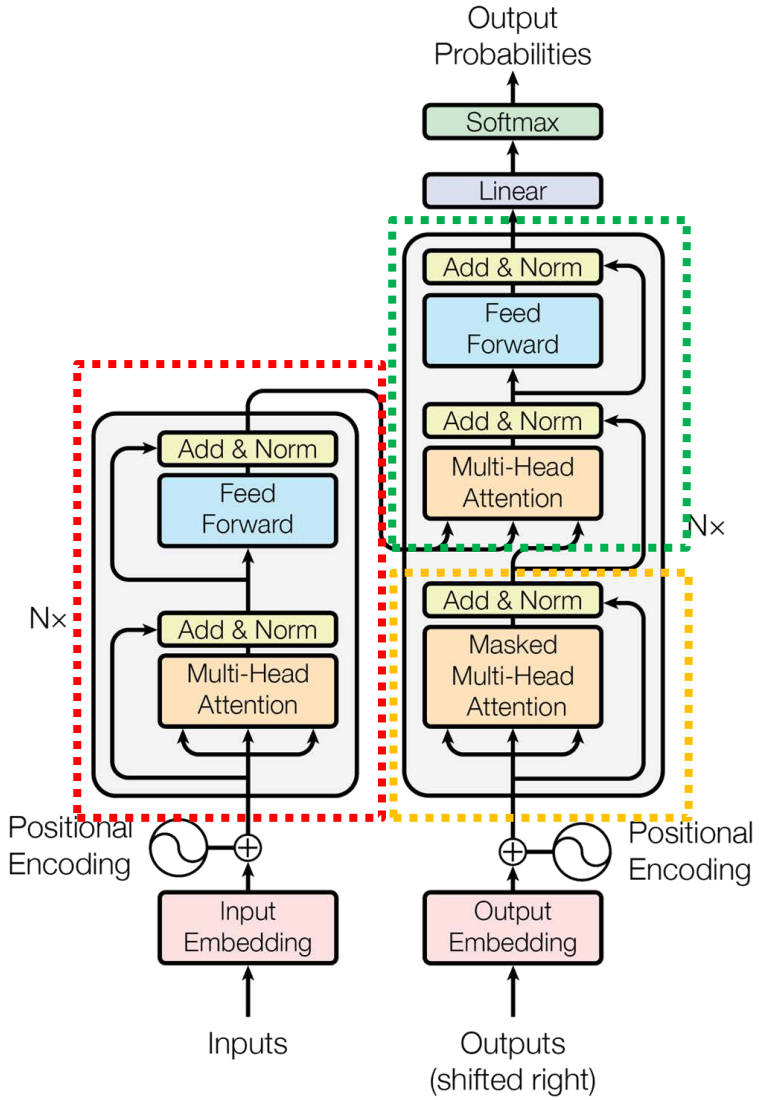
1. Encoder-Decoder
2. Attention
   - Scaled Dot-Product Attention
   - Multi-Head Attention
3. Position-wise Feed-Forward Networks
4. Positional Encoding



Decoder

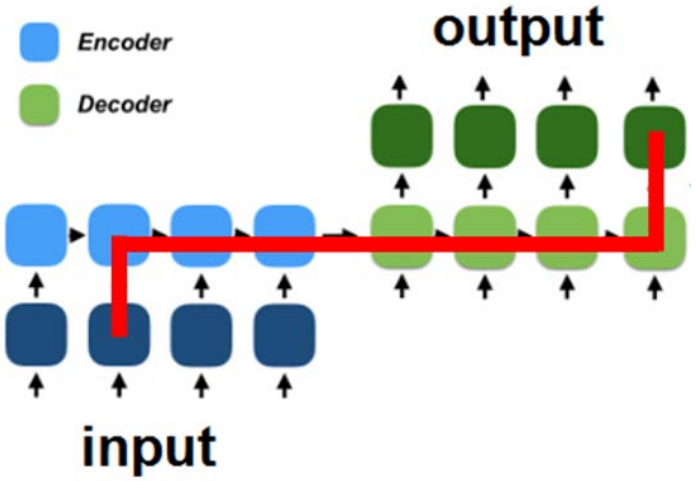
Attention

Encoder

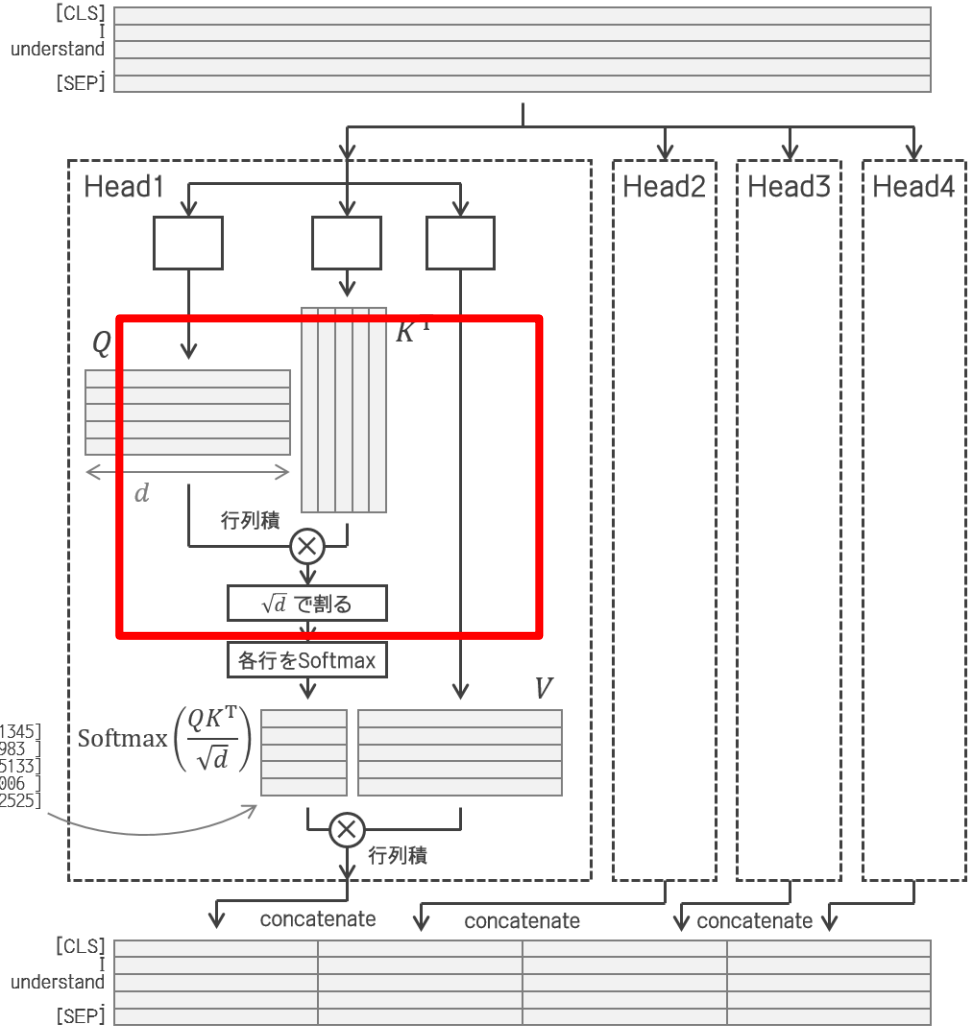


The Transformer model

1. Encoder-Decoder
2. Attention
   - Scaled Dot-Product Attention
   - Multi-Head Attention
3. Position-wise Feed-Forward Networks
4. Positional Encoding



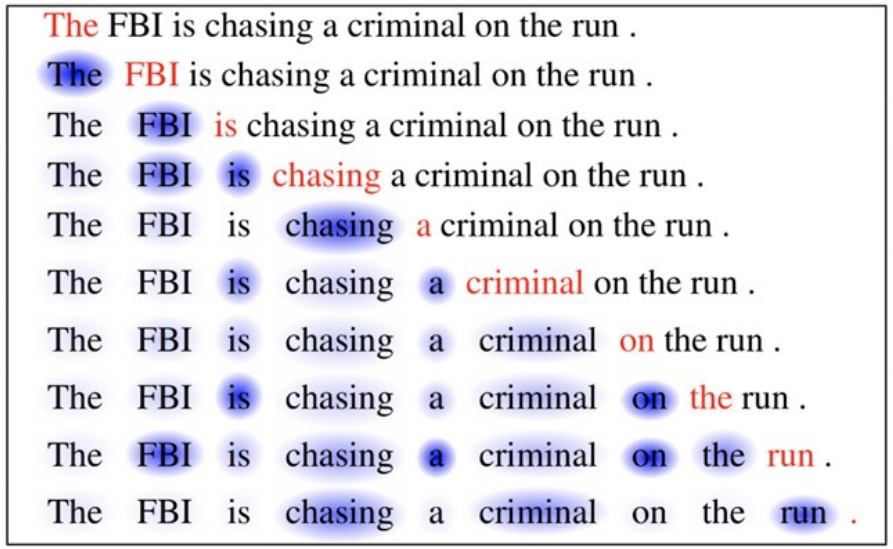
	[CLS]	I	understand	.	[SEP]
[CLS]	0.0004746	0.31851667	0.5285722	0.06112319	0.09131345
I	0.03716803	0.0598756	0.63856435	0.12849371	0.1358983
understand	0.1096666	0.12933028	0.0949462	0.2482056	0.41785133
.	0.02664849	0.07455668	0.11144524	0.14904903	0.6383006
[SEP]	0.80464935	0.01082572	0.00442281	0.07817679	0.10192525

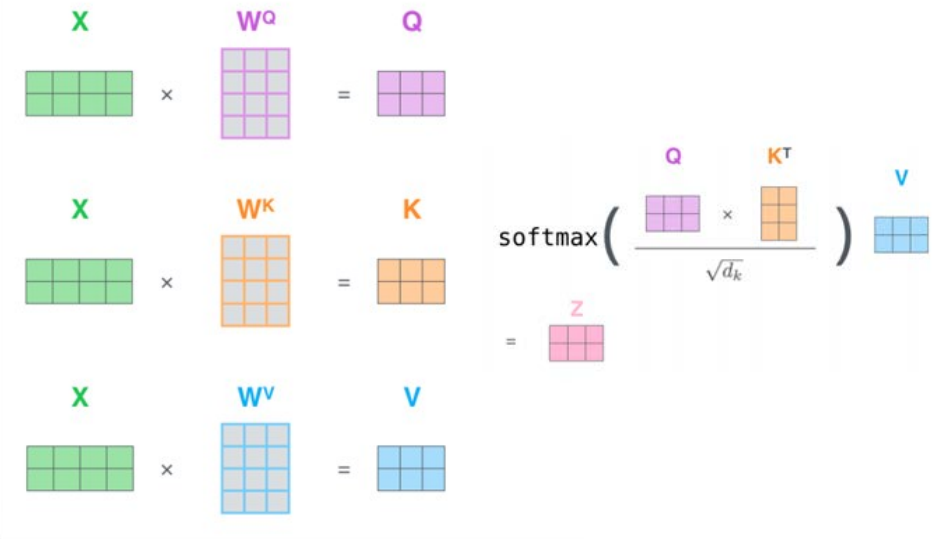


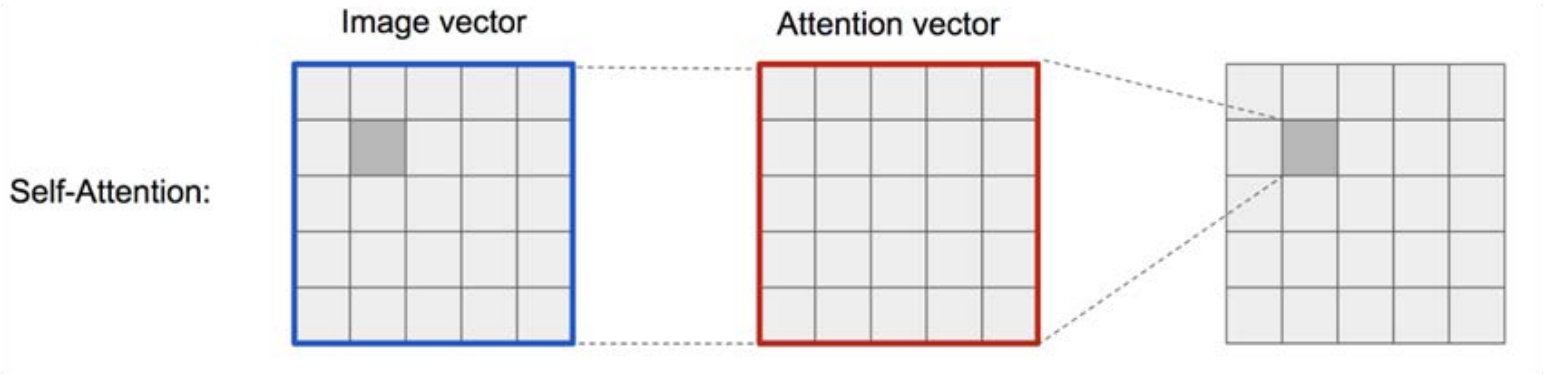
https://cookie-box.hatenablog.com/entry/2021/02/11/195822

Multi-Headed Self-Attention

The self-attention in NLP/CV field





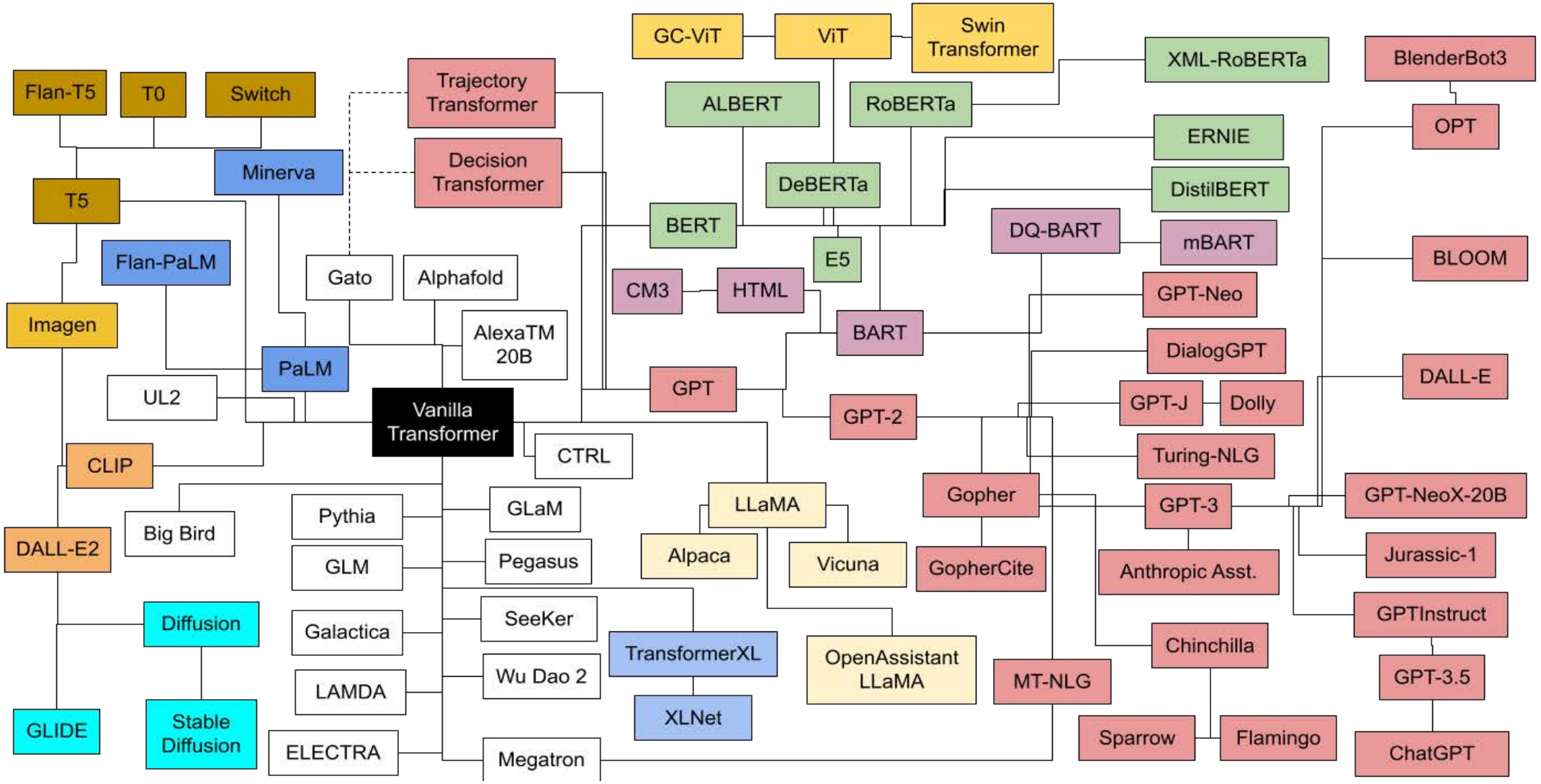


Acknowledgement from http://jalammar.github.io/illustrated-bert/

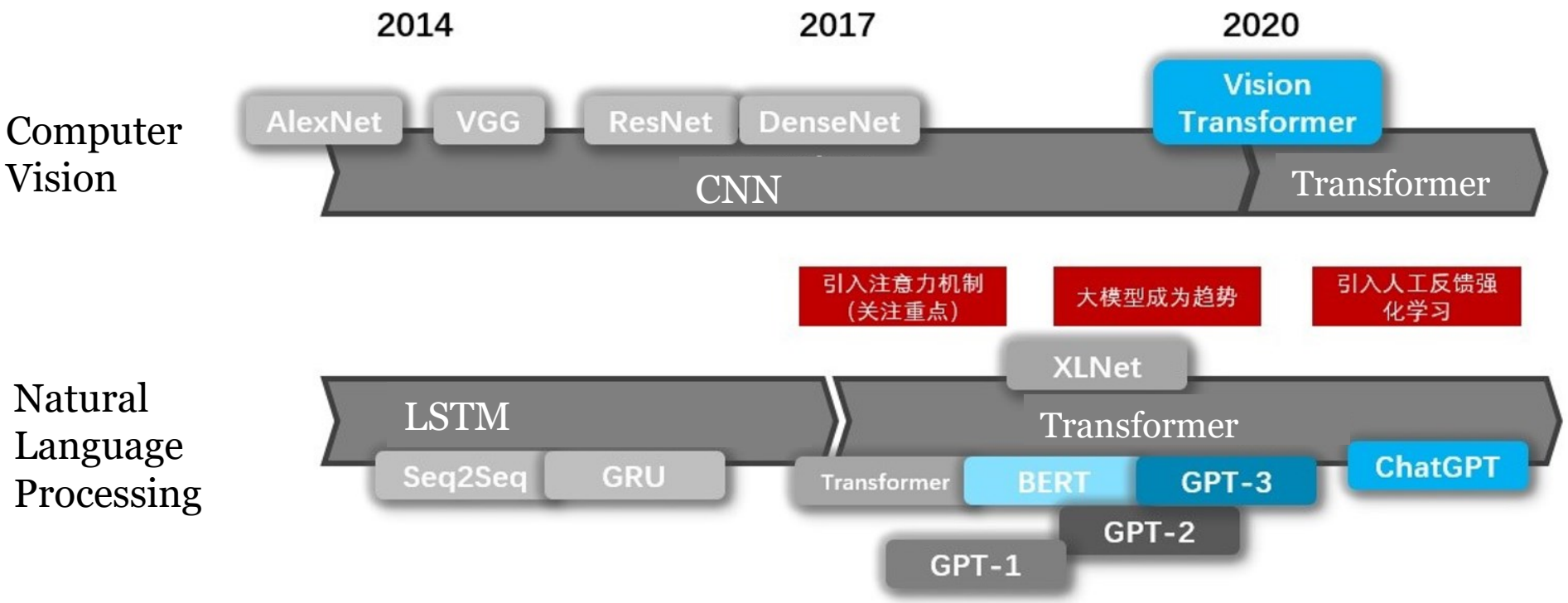
- Content -

- Sequential Modeling: from Science
- The Transformer Model
- **The modern architecture**
- Some applications on scientific sequences
- Future direction

The Development of Transformer

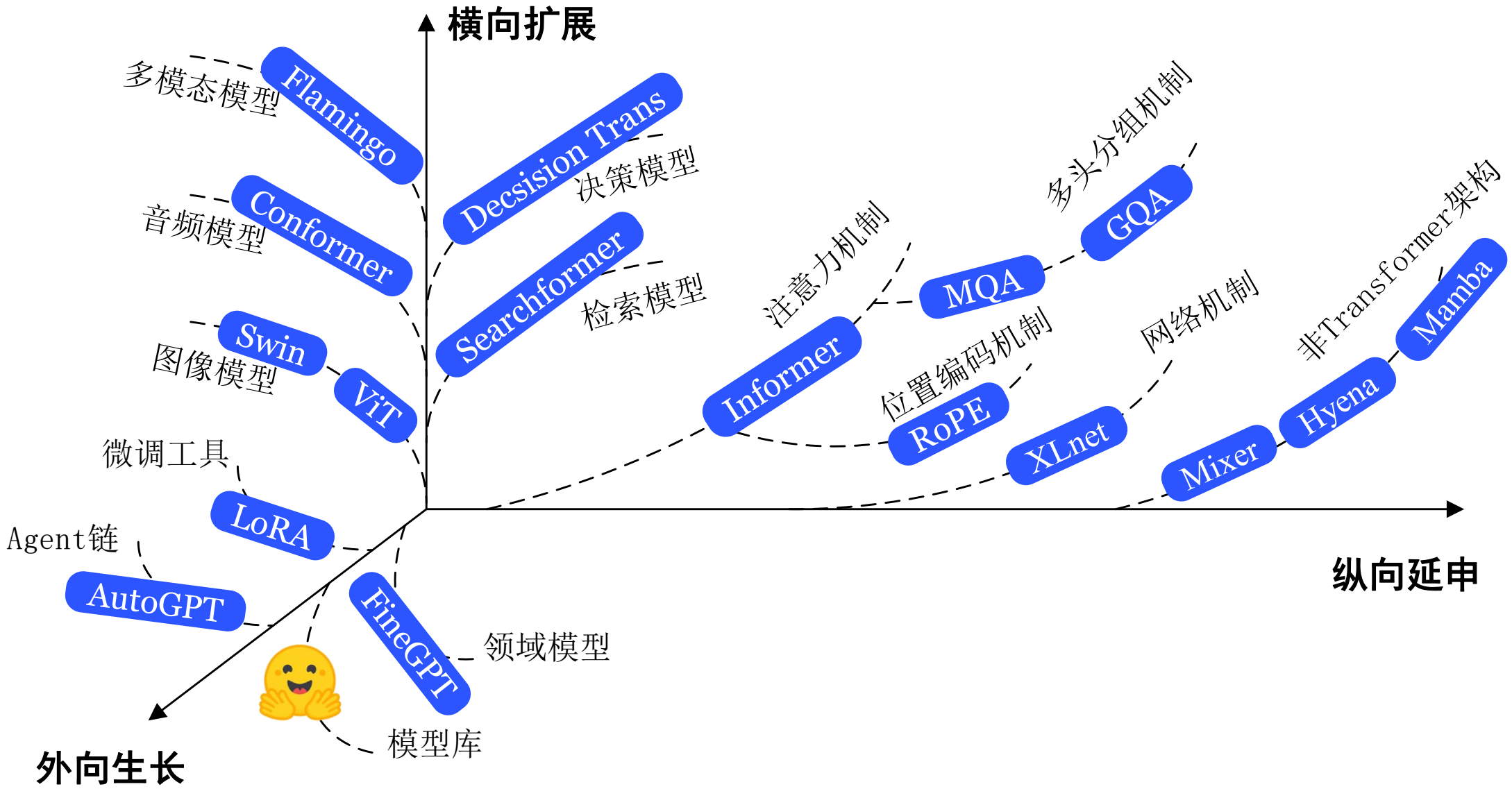


The Development of Transformer



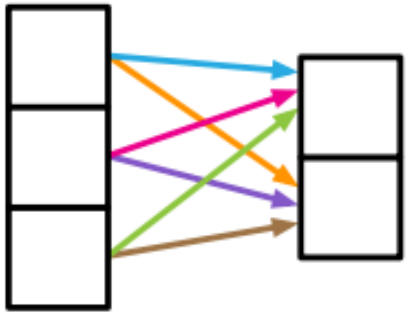
The Transformer has become the main architecture in the field of Computer Vision and Natural Language Processing.

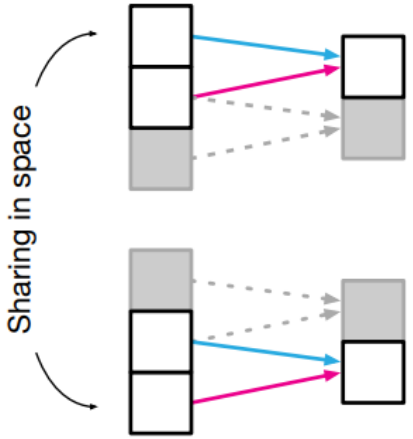
The Development of Transformer

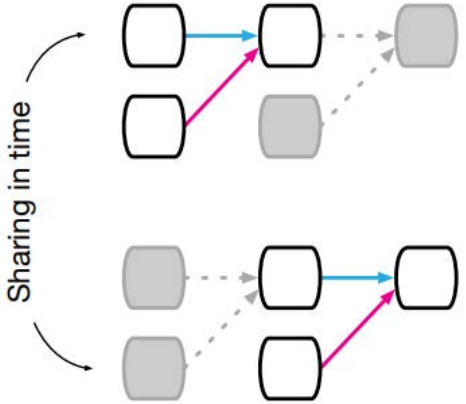


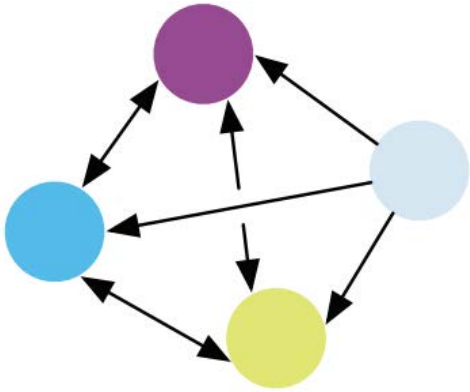
Inductive Bias and Invariance

Component	Entities	Relations	Rel. inductive bias	Invariance
Fully connected	Units	All-to-all	Weak	-
Convolutional	Grid elements	Local	Locality	Spatial translation
Recurrent	Timesteps	Sequential	Sequentiality	Time translation
Graph network	Nodes	Edges	Arbitrary	Node, edge permutations









Inductive Bias and Invariance

The attention architecture of the Transformer, compared to RNNs and CNNs, has a weaker inductive bias and relies on a large amount of training data to surpass the advantages of inductive bias.

RNN's temporal locality: It can only represent the features of adjacent sequence units, while the Attention mechanism of the Transformer can flexibly depict the relevant features of long sequences.

CNN's spatial locality and translation invariance: It is suitable for representing local image features, but requires a very deep CNN architecture to expand its receptive field. Meanwhile, the Attention mechanism of ViTal can flexibly represent features of larger visual fields of pixel blocks.

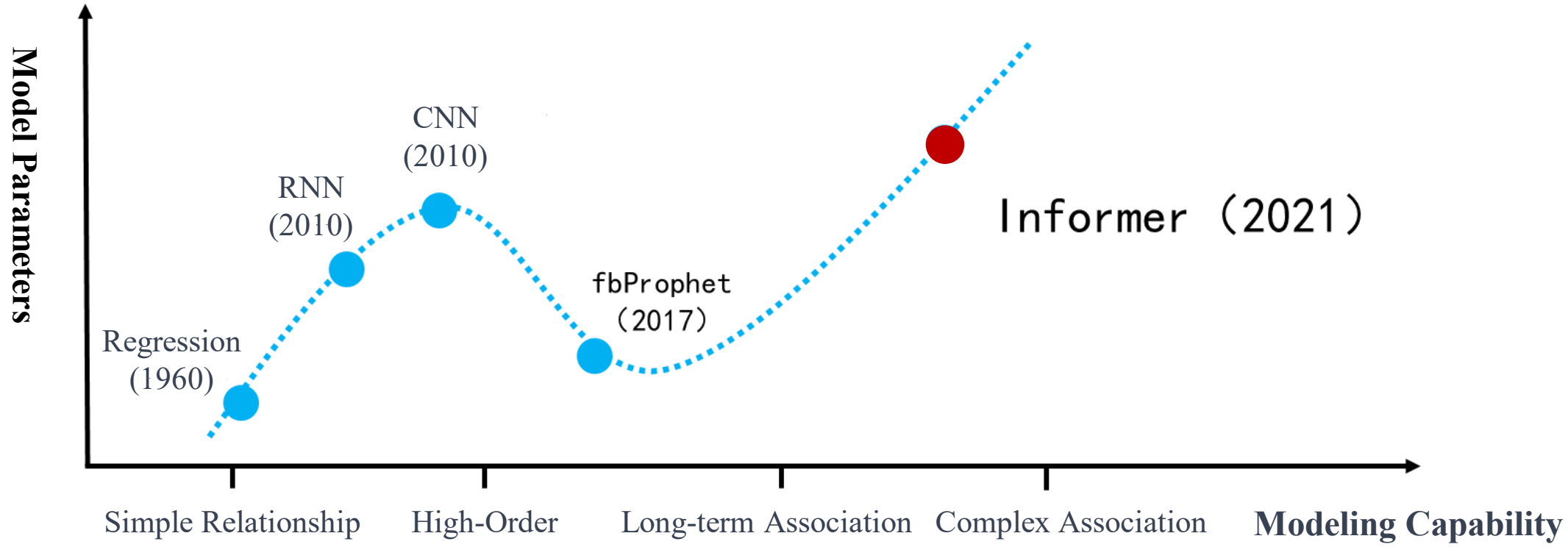
The Transformer is actually a dynamic MLP that can adjust weights through attention mechanism

Inductive Bias and Invariance

RNN	CNN	Transformer
		
number off	different group discussion	freely exchange
Limited information	**Information Loss**	**Rich Information**
Fast	**Moderate**	**Slow**

Model Foundations for Sequential Architecture

Informer (AAAI 2021), a specialized neural network model for open sequence data, provides a standard solution for long sequence modeling.

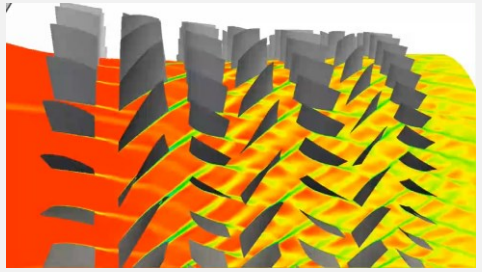


Since March 2021, tracking research on sequence processing solutions includes AutoFormer, EConformer, TSformer, FMMformer, Htransformer-1D, TCCT, TEGRU, and others.

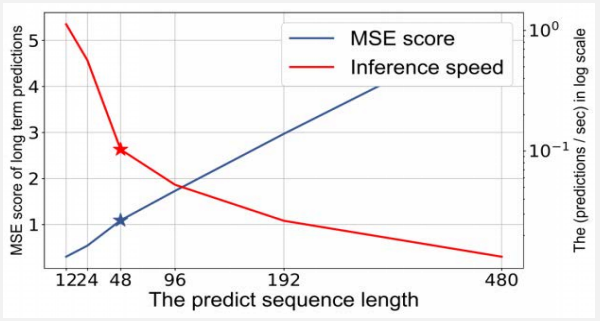
Question: Representability - Modeling Long Scientific Sequences

Problems

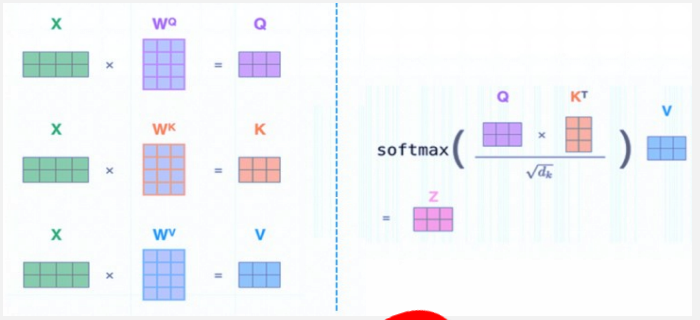
Poor accuracy in long scientific sequence modeling



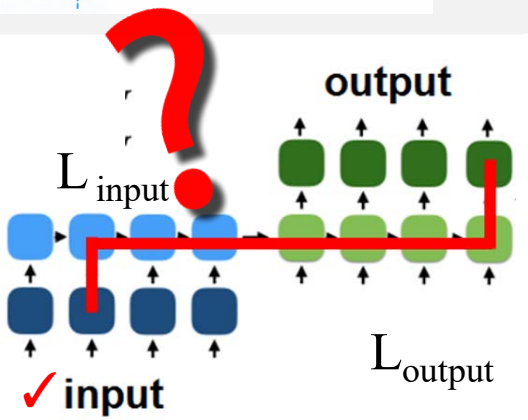
Thousands of timesteps



Capturing long-term dependencies relies on large-capacity models



	ARIMA	Prophet	LSTM	Conv	Transformer
Long-Seq Input	✓	✓	×	✓	×
Long-Seq Output	×	×	×	×	×
Complexity per Layer	O(L)	Not clear	$O(L*d^2)$	O(k*L*d)	$O(L^2*d)$
Longest Path			O(L)	$O(\log_k L)$	O(1)



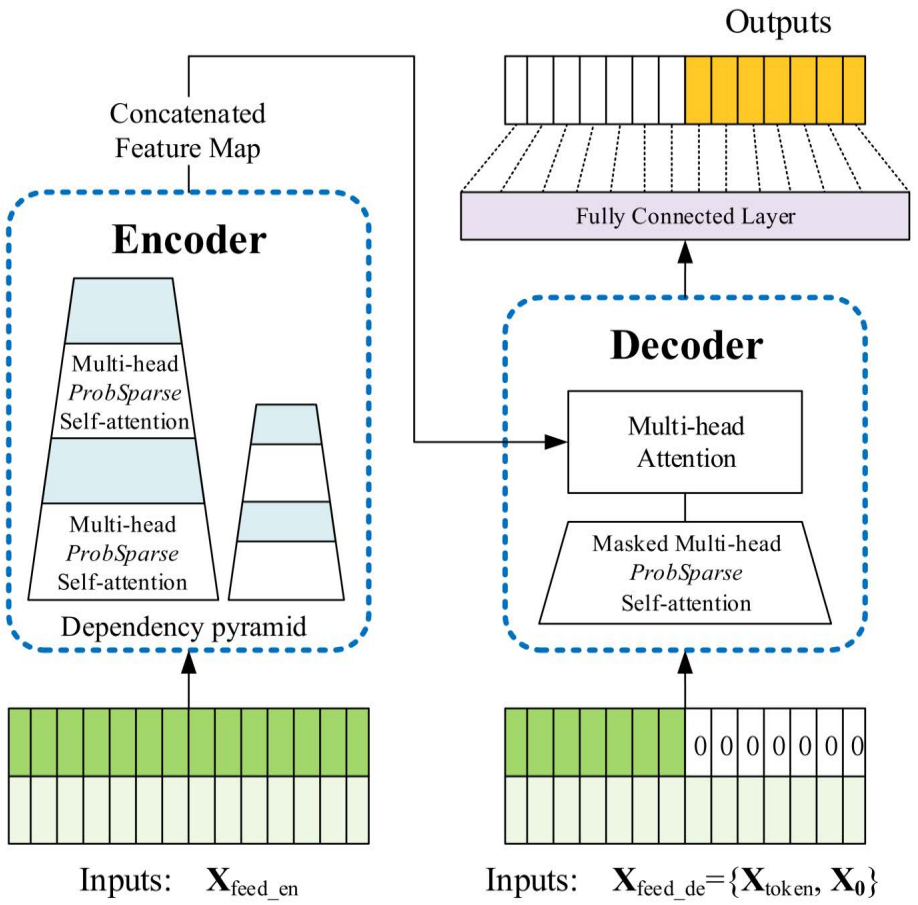
The shorter the path, the less traversal in parameter propagation within the model; the stronger the performance, the clearer the long-term dependencies are captured.

Challenges: Efficient Capture of Long Dependencies

1. **The quadratic computation of self-attention.** The atom operation of self-attention mechanism, namely canonical dot-product, causes the time complexity and memory usage per layer to be $O(L^2)$.
   [Complexity/layer]

2. **The memory bottleneck in stacking layers.** The stack of J encoder/decoder layer makes total memory usage to be $O(J \cdot L^2)$, which limits the model scalability on receiving long sequence inputs. [Long Input]

3. **The speed plunge in predicting long outputs.** The dynamic decoding of vanilla Transformer makes the inference speed as slow as RNN-based model.
   [Long Output]



An overall graph of the Informer model.

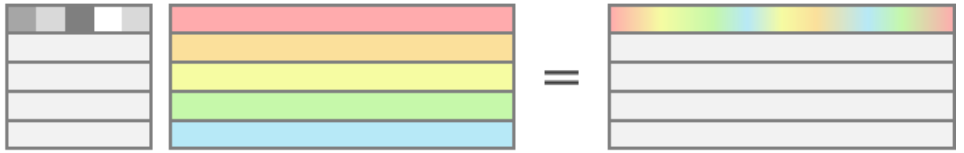
[1] Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting, **AAAI 2021 (Best paper)**

Challenge 1: Self-attention Mechanism

- Rewrite the original self-attention into the probability formulation

$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V} \quad \Rightarrow \quad \mathcal{A}(\mathbf{q}_i,\mathbf{K},\mathbf{V}) = \sum_j \frac{k(\mathbf{q}_i,\mathbf{k}_j)}{\sum_l k(\mathbf{q}_i,\mathbf{k}_l)}\mathbf{v}_j = \mathbb{E}_{p(\mathbf{k}_j|\mathbf{q}_i)}[\mathbf{v}_j]$$

$$p(k_j|q_i) = \frac{\exp(d^{-1/2} q_i \cdot k_j)}{Z_i} \rightarrow \text{Softmax}\left(\frac{QK^\top}{\sqrt{d}}\right) V$$



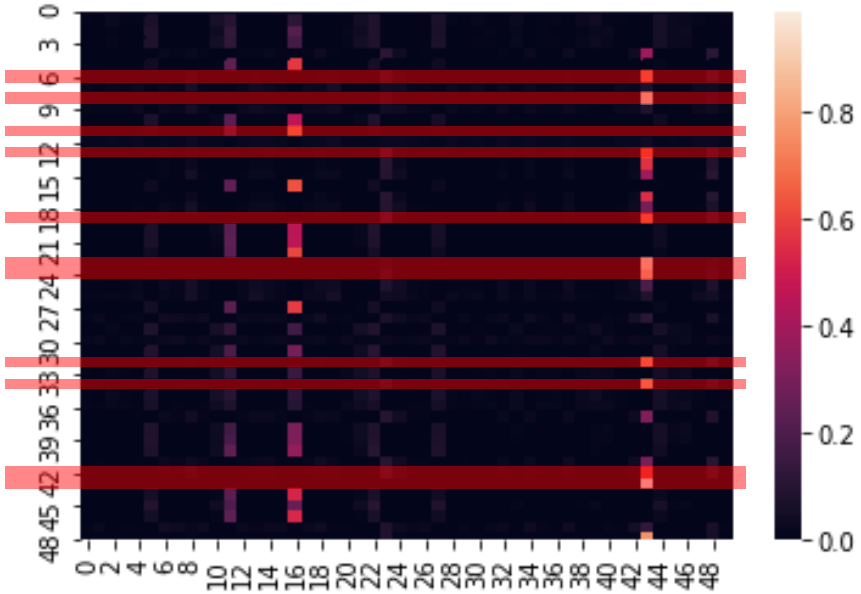
It's a new way to investigate the dot-product attention.

https://cookie-box.hatenablog.com/entry/2021/02/11/195822

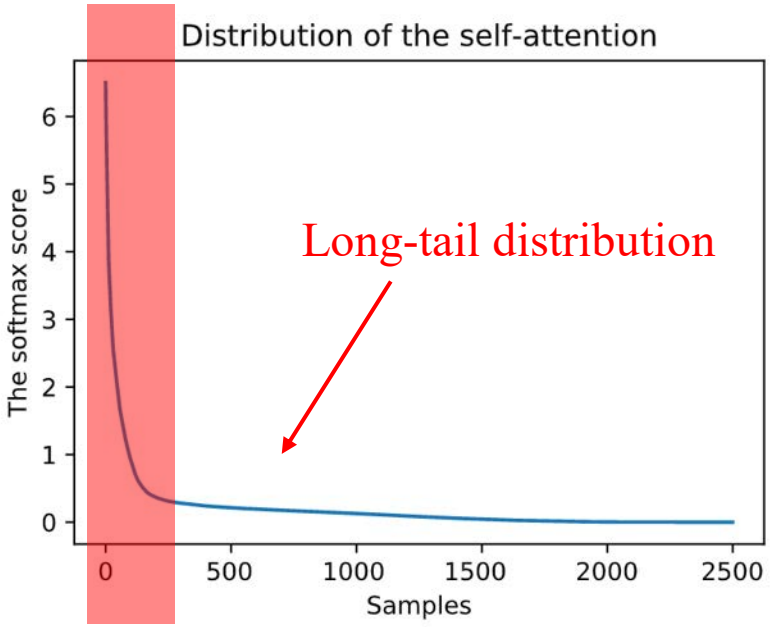
Challenge 1: Self-attention Mechanism

$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V}$$

Q

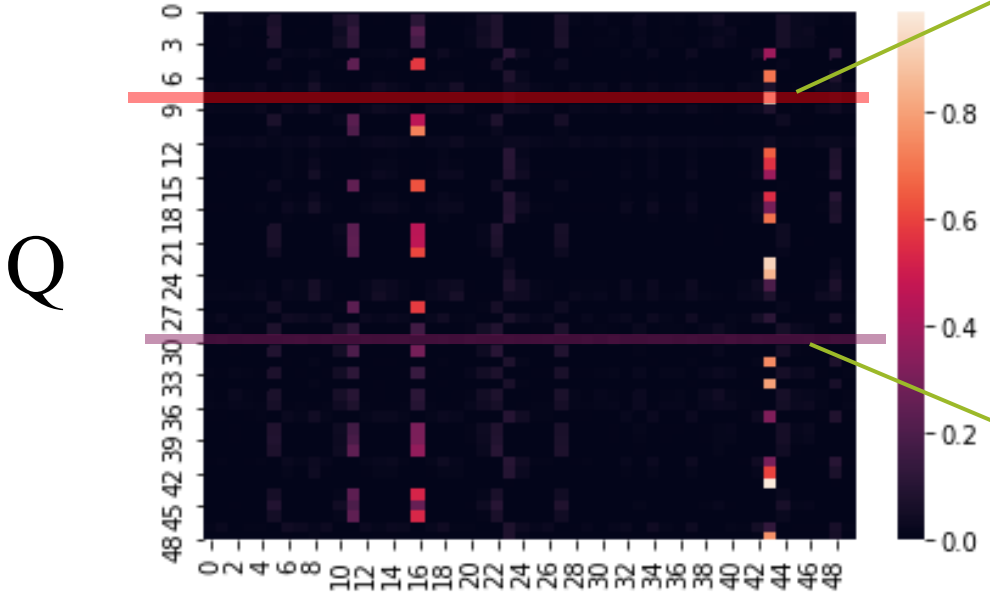


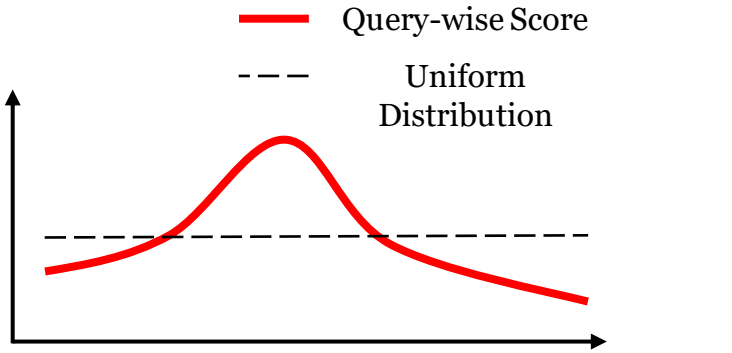
K



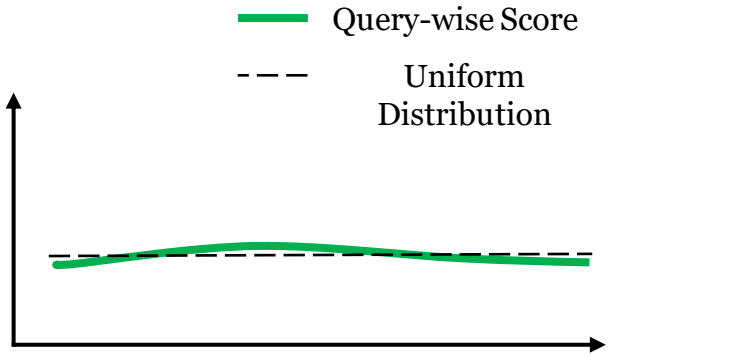
Challenge 1: Self-attention Mechanism

$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{Softmax}(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{d}})\mathbf{V}$$





The "Active" Query in Self-Attention



The "Lazy" Query in Self-Attention

Challenge 1: Self-attention Mechanism

- Rewrite the original self-attention into the probability formulation

$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{Softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V} \quad \Rightarrow \quad \mathcal{A}(\mathbf{q}_i,\mathbf{K},\mathbf{V}) = \sum_j \frac{k(\mathbf{q}_i,\mathbf{k}_j)}{\sum_l k(\mathbf{q}_i,\mathbf{k}_l)}\mathbf{v}_j = \mathbb{E}_{p(\mathbf{k}_j|\mathbf{q}_i)}[\mathbf{v}_j]$$

- Establish the measurement

Attention probability $p(\mathbf{k}_j|\mathbf{q}_i)$? Uniform probability $q(\mathbf{k}_j|\mathbf{q}_i) = \frac{1}{L_K}$

$$KL(q||p) = \sum_{j=1}^{L_K} \frac{1}{L_K} \ln \frac{1/L_K}{k(\mathbf{q}_i,\mathbf{k}_j)/\sum_l k(\mathbf{q}_i,\mathbf{k}_l)}$$

$$= \ln \sum_{l=1}^{L_K} e^{\frac{\mathbf{q}_i\mathbf{k}_l^\top}{\sqrt{d}}} - \frac{1}{L_K}\sum_{j=1}^{L_K}\frac{\mathbf{q}_i\mathbf{k}_j^\top}{\sqrt{d}} - \ln L_K$$

Dropping the constant, we define the i-th query's sparsity measurement as

$$M(\mathbf{q}_i,\mathbf{K}) = \ln \sum_{j=1}^{L_K} e^{\frac{\mathbf{q}_i\mathbf{k}_j^\top}{\sqrt{d}}} - \frac{1}{L_K}\sum_{j=1}^{L_K}\frac{\mathbf{q}_i\mathbf{k}_j^\top}{\sqrt{d}}$$

Challenge 1: Self-attention Mechanism

- Rewrite the original self-attention into the probability formulation

$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V} \quad \Rightarrow \quad \mathcal{A}(\mathbf{q}_i,\mathbf{K},\mathbf{V}) = \sum_j \frac{k(\mathbf{q}_i,\mathbf{k}_j)}{\sum_l k(\mathbf{q}_i,\mathbf{k}_l)}\mathbf{v}_j = \mathbb{E}_{p(\mathbf{k}_j|\mathbf{q}_i)}[\mathbf{v}_j]$$

- Establish the measurement

$$M(\mathbf{q}_i,\mathbf{K}) = \ln \sum_{j=1}^{L_K} e^{\frac{\mathbf{q}_i\mathbf{k}_j^\top}{\sqrt{d}}} - \frac{1}{L_K}\sum_{j=1}^{L_K}\frac{\mathbf{q}_i\mathbf{k}_j^\top}{\sqrt{d}}$$

- Define *ProbSparse* self-attention

Based on proposed measurement, we have the *ProbSparse* Self-attention by allowing each key only to attend to the *u* dominate queries → The "Active" queries

$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}(\frac{\overline{\mathbf{Q}}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V}$$

where $\overline{\mathbf{Q}}$ is a sparse matrix of the same size of **q** and it only contains the Top-*u* queries under the sparsity measurement $M(\mathbf{q},\mathbf{K})$.

Challenge 1: Self-attention Mechanism

- Rewrite the original self-attention into probability formulation
- Establish the measurement
- Define *ProbSparse* self-attention

- The measurement have to calculate each dot-product pairs, which still requires $O(L^2)$ computation. And the log-sum-exp fun has the numerical stability issue.

$$M(\mathbf{q}_i, \mathbf{K}) = \ln \sum_{j=1}^{L_K} e^{\frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}}} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}} \quad \text{Replace} \quad \max_j \{ \frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}} \}$$

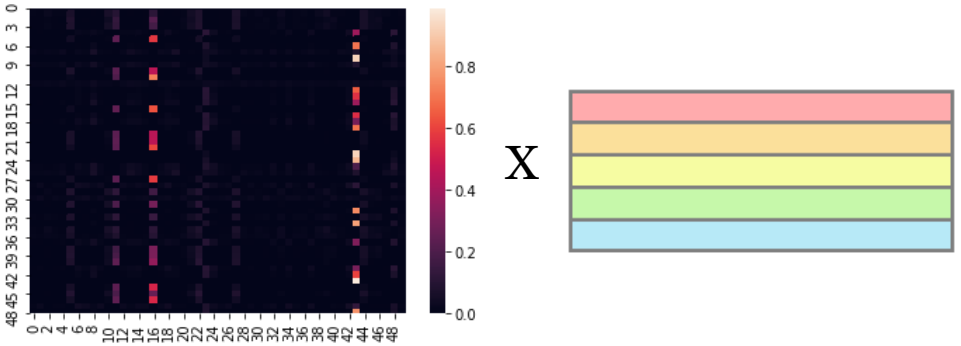
Lemma 1. *For each query* $\mathbf{q}_i \in \mathbb{R}^d$ *and* $\mathbf{k}_j \in \mathbb{R}^d$ *in the keys set* $\mathbf{K}$, *we have the following lower and upper bounds*

$$\ln L_K \leq M(\mathbf{q}_i, \mathbf{K}) \leq \max_j \{ \frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}} \} - \frac{1}{L_K} \sum_{j=1}^{L_K} \{ \frac{\mathbf{q}_i \mathbf{k}_j^\top}{\sqrt{d}} \} + \ln L_K .$$

Especially when $\mathbf{q}_i \in \mathbf{K}$, *it also holds.*

Challenge 1: Self-attention Mechanism

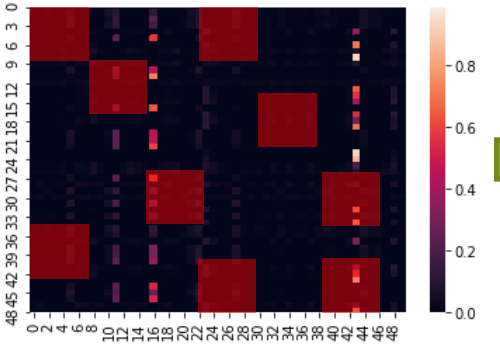
Transformer (2017):

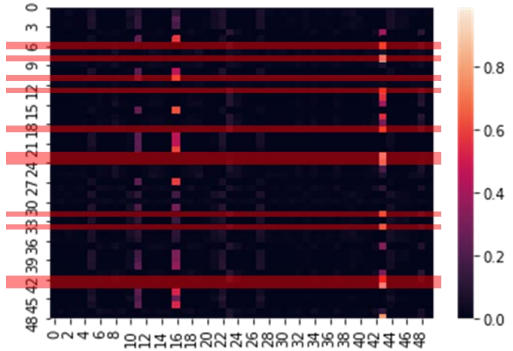


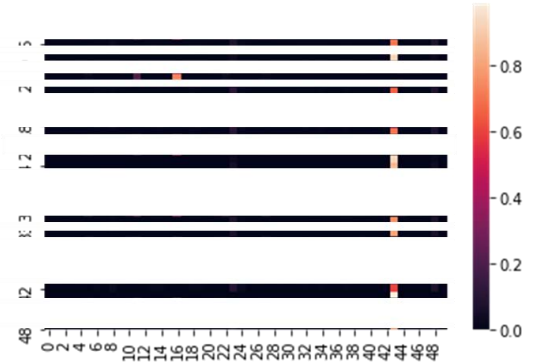
X

O(L log L) $O(L^2)$ ✓

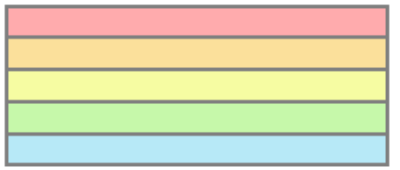
Informer (2021):







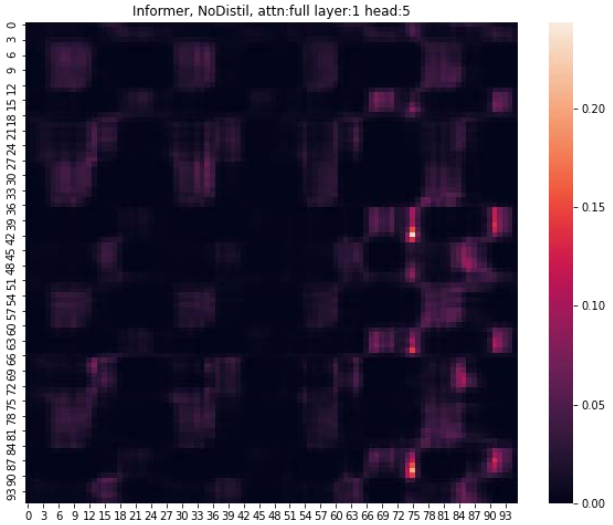
X

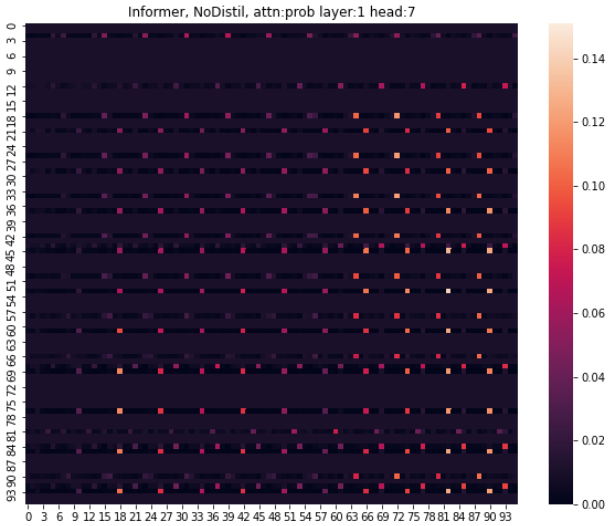


Sample (L log L) dot-product pairs Use M select *u* queries Calculate Probspare Attention

Challenge 2: Self-attention Distilling Operation

Layer 2

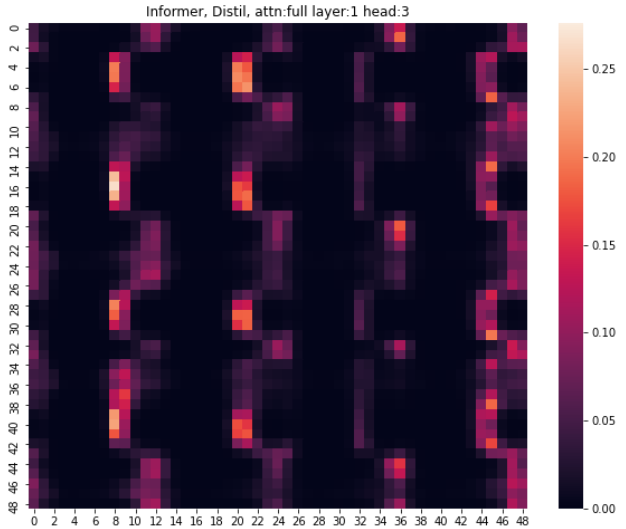


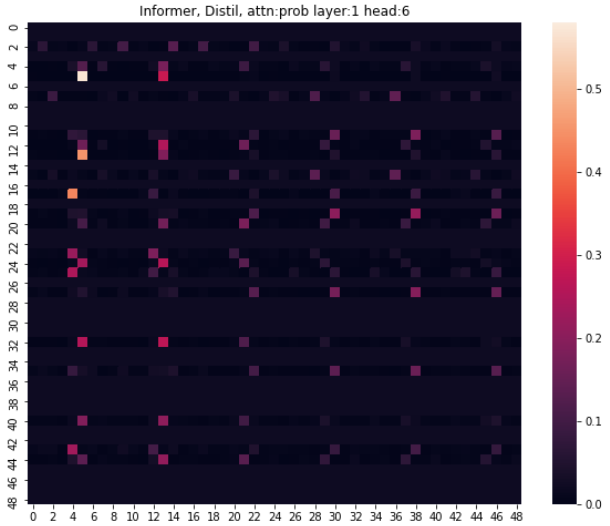


reduce

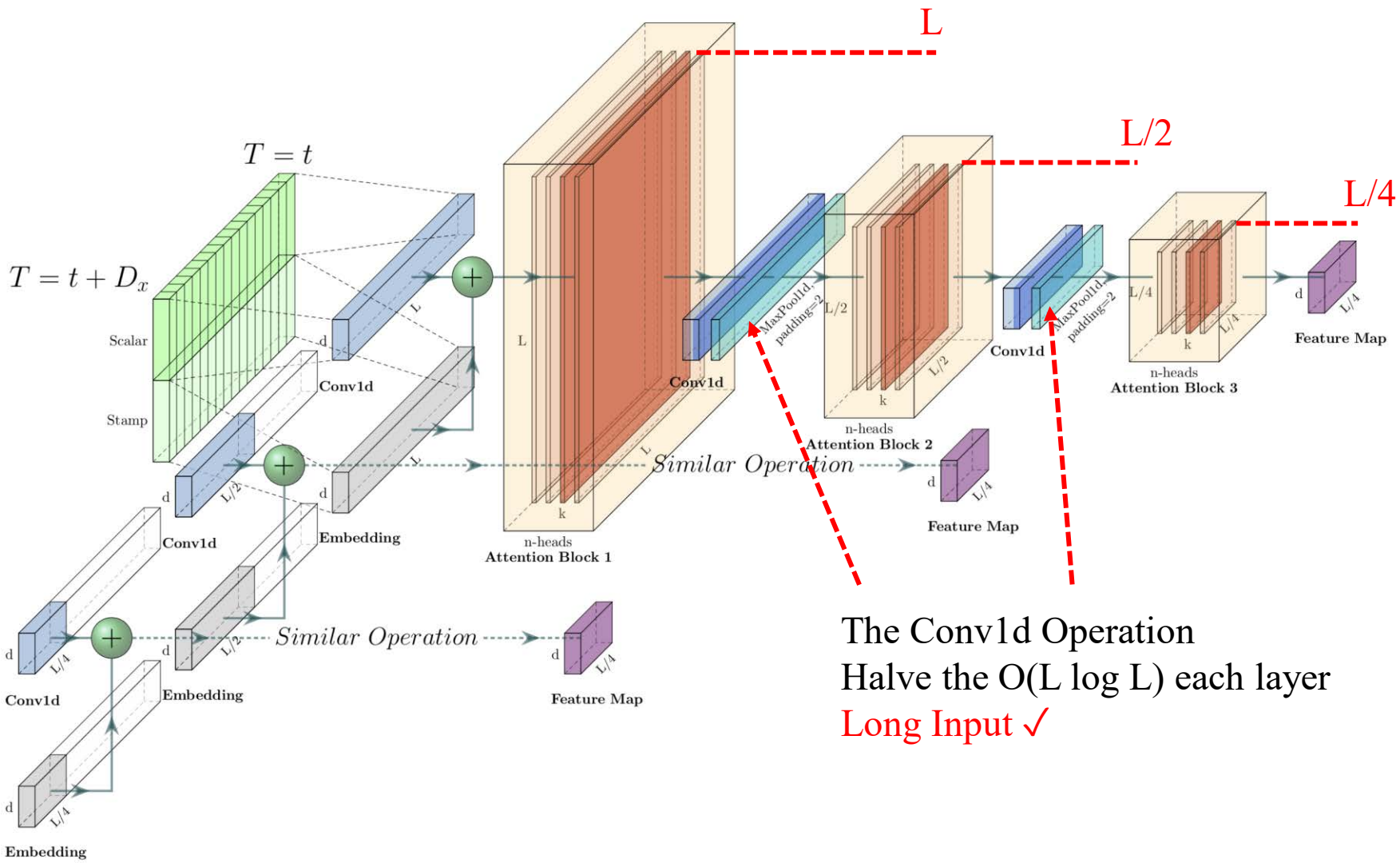
reduce

Layer 2



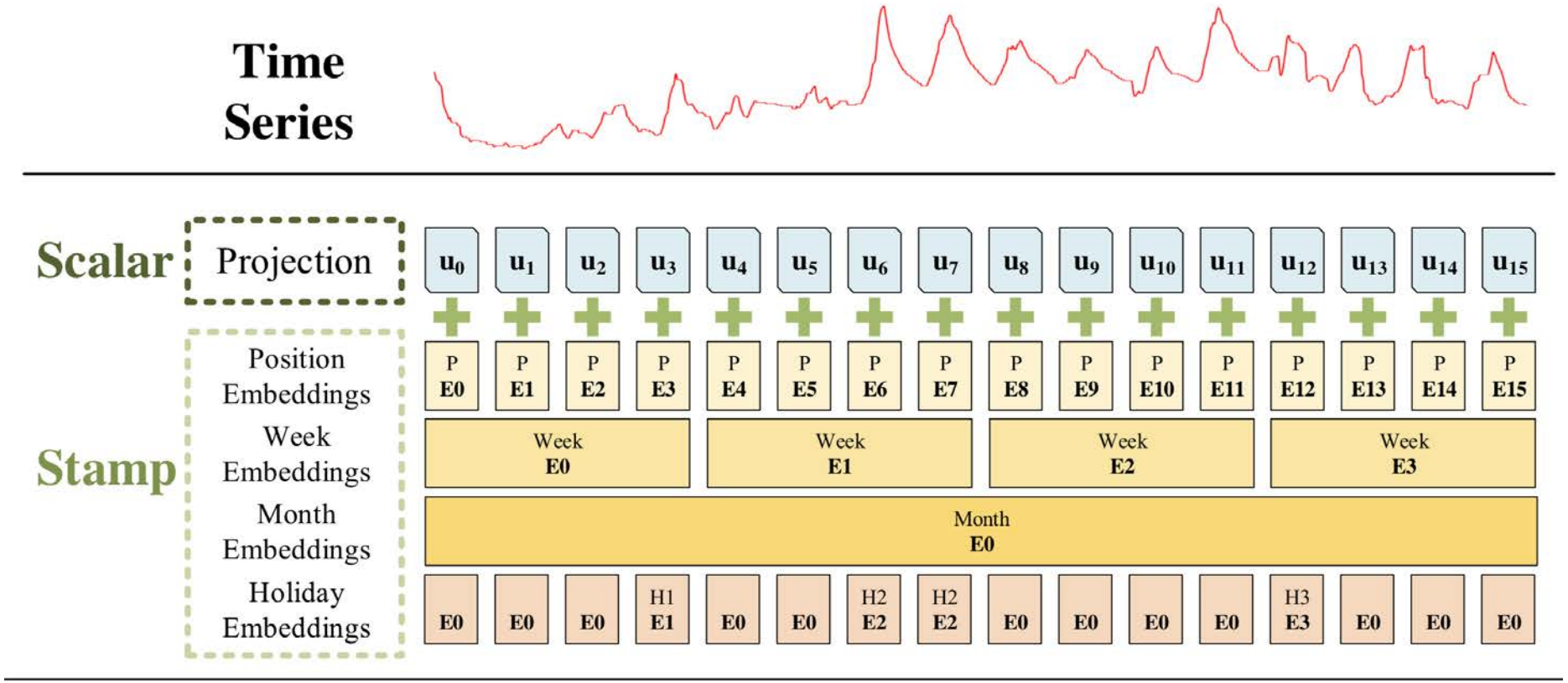


Challenge 2: Self-attention Distilling Operation



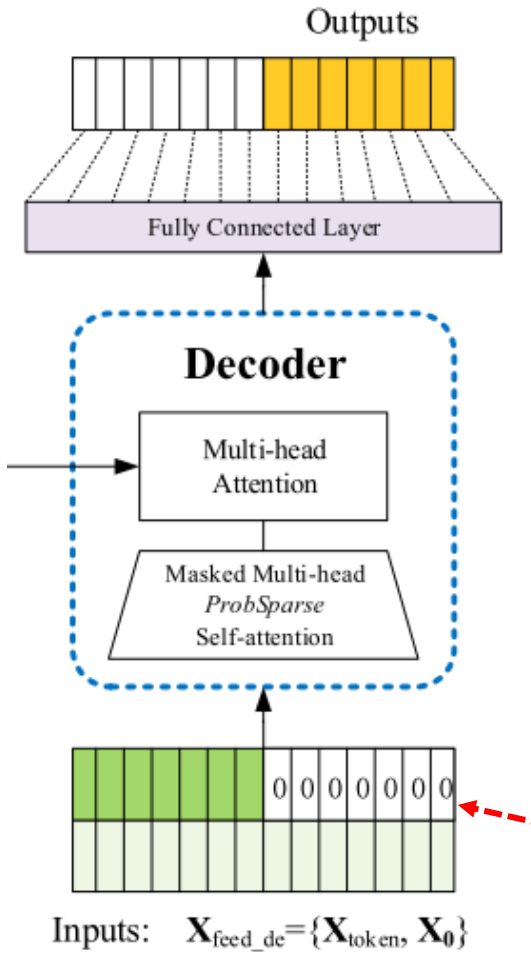
The Conv1d Operation
Halve the O(L log L) each layer
Long Input ✓

Challenge 3: Generative-style Decoder



$$\mathcal{X}^t_{\text{feed}[i]} = \alpha \mathbf{u}^t_i + \mathbf{PE}_{(D_x \times (t-1)+i,\ )} + \sum_p [\mathbf{SE}_{(D_x \times (t-1)+i)}]_p$$

Challenge 3: Generative-style Decoder



Start token is an efficient technique in NLP "dynamic decoding", especially for per-training model, and we extend it into a generative way. Instead of choosing a specific flag as the token, we sample a "shorter" long sequence in input sequence, which is an earlier slice before output sequence. Take predicting 480 points as an example (5-day prediction in the ETT dataset), we will take the known 5 days before this time sequence as "start-token", and feed the generative-style inference decoder with them.

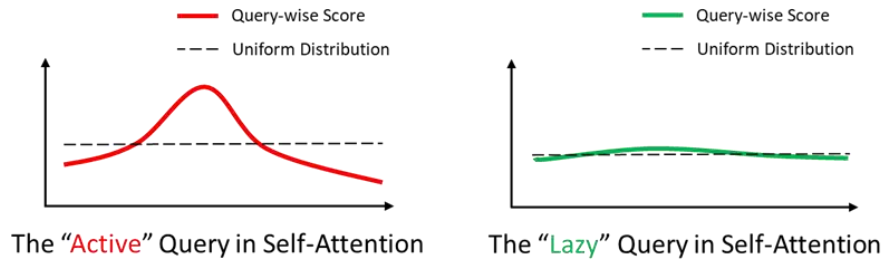
$$\mathbf{X}^t_{feed\_de} = \text{Concat}(\mathbf{X}^t_{token}, \mathbf{X}^t_{\mathbf{0}}) \in \mathbb{R}^{(L_{token}+L_y)\times d_{model}}$$

Long Output ✓

Overview of Informer

Challenge 1 [Complexity/layer]

Problem: The quadratic computation of self-attention.
Solution: ***ProbSparse* Attention**



The "Active" Query in Self-Attention

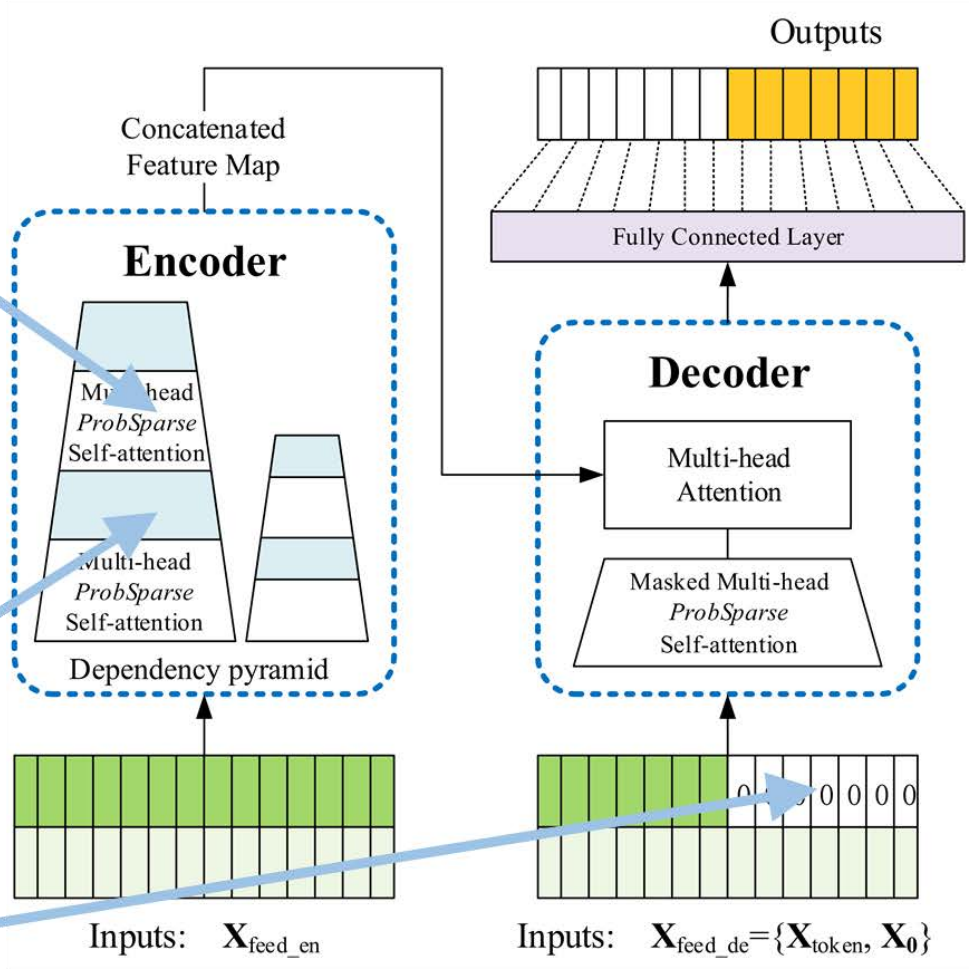
The "Lazy" Query in Self-Attention

Challenge 2 [Long Input]

Problem: The memory bottleneck in stacking layers.
Solution: **Self-attention Distilling**

Challenge 3 [Long Output]

Problem: The speed plunge in predicting long outputs.
Solution: **Generative-style Decoder**



An overall graph of the Informer model.

Experiment settings

Datasets

- ETT (Electricity Transformer Temperature): 2 stations, 2 years, every 15 minutes. Train/Val/Test is 1st year / 4 months / 8 months.
- ECL (Electricity Consuming Load): 1 country, 2 years, every 1 hour. Train/Val/Test is 15 months /3 months / 6 months.
- Weather: 1600 US locations, 4 years, every 1 hour. Train/Val/Test is 28 months / 10 months / 10 months.

Baselines

Time-series methods: ARIMA, DeepAR, Prophet, LSTMa, LSTnet
Transformer-based methods: vanilla Transformer, Reformer, LogSparse Transformer

Metrics

Mean Absolute Error (MAE), Mean Squared Error (MSE).

Platform

All methods are run on one single Nvidia V100 GPU.

(1) Univariate Time-series Forecasting

Methods		Informer		Informer†		LogTrans		Reformer		LSTMa		DeepAR		ARIMA		Prophet	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh_1$	24	0.098	0.247	**0.092**	**0.246**	0.103	0.259	0.222	0.389	0.114	0.272	0.107	0.280	0.108	0.284	0.115	0.275
	48	**0.158**	**0.319**	0.161	0.322	0.167	0.328	0.284	0.445	0.193	0.358	0.162	0.327	0.175	0.424	0.168	0.330
	168	**0.183**	**0.346**	0.187	0.355	0.207	0.375	1.522	1.191	0.236	0.392	0.239	0.422	0.396	0.504	1.224	0.763
	336	0.222	0.387	**0.215**	**0.369**	0.230	0.398	1.860	1.124	0.590	0.698	0.445	0.552	0.468	0.593	1.549	1.820
	720	0.269	0.435	**0.257**	**0.421**	0.273	0.463	2.112	1.436	0.683	0.768	0.658	0.707	0.659	0.766	2.735	3.253
$ETTh_2$	24	**0.093**	**0.240**	0.099	0.241	0.102	0.255	0.263	0.437	0.155	0.307	0.098	0.263	3.554	0.445	0.199	0.381
	48	**0.155**	**0.314**	0.159	0.317	0.169	0.348	0.458	0.545	0.190	0.348	0.163	0.341	3.190	0.474	0.304	0.462
	168	**0.232**	**0.389**	0.235	0.390	0.246	0.422	1.029	0.879	0.385	0.514	0.255	0.414	2.800	0.595	2.145	1.068
	336	0.263	**0.417**	**0.258**	0.423	0.267	0.437	1.668	1.228	0.558	0.606	0.604	0.607	2.753	0.738	2.096	2.543
	720	**0.277**	**0.431**	0.285	0.442	0.303	0.493	2.030	1.721	0.640	0.681	0.429	0.580	2.878	1.044	3.355	4.664
$ETTm_1$	24	**0.030**	**0.137**	0.034	0.160	0.065	0.202	0.095	0.228	0.121	0.233	0.091	0.243	0.090	0.206	0.120	0.290
	48	0.069	0.203	**0.066**	**0.194**	0.078	0.220	0.249	0.390	0.305	0.411	0.219	0.362	0.179	0.306	0.133	0.305
	96	0.194	**0.372**	**0.187**	0.384	0.199	0.386	0.920	0.767	0.287	0.420	0.364	0.496	0.272	0.399	0.194	0.396
	288	**0.401**	0.554	0.409	**0.548**	0.411	0.572	1.108	1.245	0.524	0.584	0.948	0.795	0.462	0.558	0.452	0.574
	672	**0.512**	**0.644**	0.519	0.665	0.598	0.702	1.793	1.528	1.064	0.873	2.437	1.352	0.639	0.697	2.747	1.174
Weather	24	**0.117**	**0.251**	0.119	0.256	0.136	0.279	0.231	0.401	0.131	0.254	0.128	0.274	0.219	0.355	0.302	0.433
	48	**0.178**	0.318	0.185	**0.316**	0.206	0.356	0.328	0.423	0.190	0.334	0.203	0.353	0.273	0.409	0.445	0.536
	168	**0.266**	**0.398**	0.269	0.404	0.309	0.439	0.654	0.634	0.341	0.448	0.293	0.451	0.503	0.599	2.441	1.142
	336	**0.297**	**0.416**	0.310	0.422	0.359	0.484	1.792	1.093	0.456	0.554	0.585	0.644	0.728	0.730	1.987	2.468
	720	**0.359**	**0.466**	0.361	0.471	0.388	0.499	2.087	1.534	0.866	0.809	0.499	0.596	1.062	0.943	3.859	1.144
E_L	48	0.239	0.359	0.238	0.368	0.280	0.429	0.971	0.884	0.493	0.539	**0.204**	**0.357**	0.879	0.764	0.524	0.595
	168	0.447	0.503	0.442	0.514	0.454	0.529	1.671	1.587	0.723	0.655	**0.315**	**0.436**	1.032	0.833	2.725	1.273
	336	0.489	0.528	0.501	0.552	0.514	0.563	3.528	2.196	1.212	0.898	**0.414**	**0.519**	1.136	0.876	2.246	3.077
	720	**0.540**	**0.571**	0.543	0.578	0.558	0.609	4.891	4.047	1.511	0.966	0.563	0.595	1.251	0.933	4.243	1.415
	960	**0.582**	**0.608**	0.594	0.638	0.624	0.645	7.019	5.105	1.545	1.006	0.657	0.683	1.370	0.982	6.901	4.264
Count		32		12		0		0		0		6		0		0	

Informer (32)

Transformer (12)

- The *Informer* greatly improves the inference performance.
- The *Informer* and its canonical degradation *Informer*+ show comparable performance and *Informer* beats *Informer*+ mostly in wining-counts, i.e. 32>12.
- The *Informer* model shows significantly better results than time-series models.
- Our proposed methods achieve better results than Reformer and LogTrans.

(2) Multivariate Time-series Forecasting

Methods		Informer		Informer†		LogTrans		Reformer		LSTMa		LSTnet	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh_1$	24	**0.577**	**0.549**	0.620	0.57	0.686	0.604	0.991	0.754	0.650	0.624	1.293	0.901
	48	**0.685**	**0.625**	0.692	0.67	0.766	0.757	1.313	0.906	0.702	0.675	1.456	0.960
	168	**0.931**	**0.752**	0.947	0.79	1.002	0.846	1.824	1.138	1.212	0.867	1.997	1.214
	336	1.128	0.873	**1.094**	**0.81**	1.362	0.952	2.117	1.280	1.424	0.994	2.655	1.369
	720	**1.215**	**0.896**	1.241	0.91	1.397	1.291	2.415	1.520	1.960	1.322	2.143	1.380
$ETTh_2$	24	**0.720**	**0.665**	0.753	0.72	0.828	0.750	1.531	1.613	1.143	0.813	2.742	1.457
	48	**1.457**	**1.001**	1.461	1.07	1.806	1.034	1.871	1.735	1.671	1.221	3.567	1.687
	168	3.489	**1.515**	3.485	1.61	4.070	1.681	4.660	1.846	4.117	1.674	**3.242**	2.513
	336	2.723	1.340	2.626	**1.28**	3.875	1.763	4.028	1.688	3.434	1.549	**2.544**	2.591
	720	**3.467**	**1.473**	3.548	1.49	3.913	1.552	5.381	2.015	3.963	1.788	4.625	3.709
$ETTm_1$	24	0.323	**0.369**	**0.306**	0.37	0.419	0.412	0.724	0.607	0.621	0.629	1.968	1.170
	48	0.494	0.503	**0.465**	**0.47**	0.507	0.583	1.098	0.777	1.392	0.939	1.999	1.215
	96	**0.678**	0.614	0.681	**0.61**	0.768	0.792	1.433	0.945	1.339	0.913	2.762	1.542
	288	**1.056**	**0.786**	1.162	0.87	1.462	1.320	1.820	1.094	1.740	1.124	1.257	2.076
	672	**1.192**	**0.926**	1.231	1.10	1.669	1.461	2.187	1.232	2.736	1.555	1.917	2.941
Weather	24	**0.335**	**0.381**	0.349	0.39	0.435	0.477	0.655	0.583	0.546	0.570	0.615	0.545
	48	0.395	0.459	**0.386**	**0.43**	0.426	0.495	0.729	0.666	0.829	0.677	0.660	0.589
	168	**0.608**	**0.567**	0.613	0.58	0.727	0.671	1.318	0.855	1.038	0.835	0.748	0.647
	[illegible]	**0.70**	**0.620**	0.707	0.63	0.754	0.670	1.930	1.167	1.657	1.059	0.782	0.683
	720	**0.85**	**0.731**	0.834	0.74	0.885	0.773	2.726	1.575	1.536	1.109	0.851	0.757
E/L	48	0.344	**0.393**	**0.334**	0.39	0.355	0.418	1.404	0.999	0.486	0.572	0.369	0.445
	168	0.368	0.424	**0.353**	**0.42**	0.368	0.432	1.515	1.069	0.574	0.602	0.394	0.476
	336	0.381	**0.431**	0.381	0.43	**0.373**	0.439	1.601	1.104	0.886	0.795	0.419	0.477
	720	[illegible]	[illegible]	**0.391**	**0.43**	0.409	0.454	2.009	1.170	1.676	1.095	0.556	0.565
	960	**0.460**	**0.548**	0.492	0.55	0.477	0.589	2.141	1.387	1.591	1.128	0.605	0.599
Count		33		14		1		0		0		2	

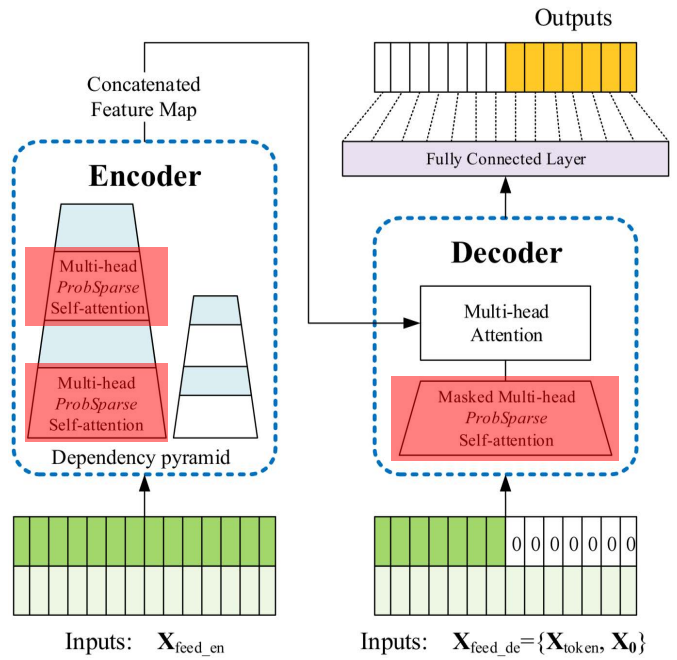
Informer (33)

Transformer (14)

- The *Informer* greatly improves the inference performance.
- The *Informer* and its canonical degradation *Informer*+ show comparable performance and *Informer* beats *Informer*$^+$ mostly in wining-counts, i.e. 33>14.
- The previous conclusion in multivariate case holds.

(3) Ablation study: The ProbSparse Attention

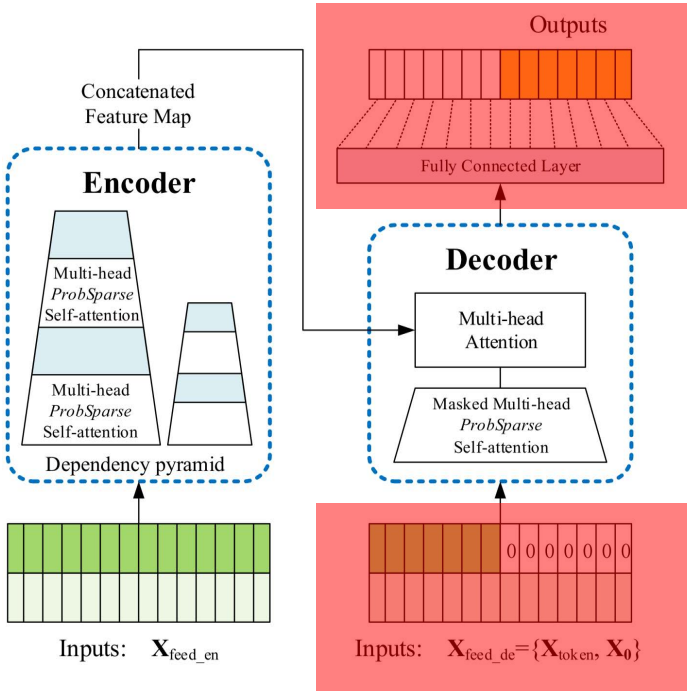
Prediction length		336			720		
Encoder's input		336	720	1440	720	1440	2880
Informer	MSE	0.249	0.225	0.216	0.271	0.261	0.257
	MAE	0.393	0.384	0.376	0.435	0.431	0.422
Informer†	MSE	0.241	0.214	-	0.259	-	-
	MAE	0.383	0.371	-	0.423	-	-
LogTrans	MSE	0.263	0.231	-	0.273	-	-
	MAE	0.418	0.398	-	0.463	-	-
Reformer	MSE	1.875	1.865	1.861	2.243	2.174	2.113
	MAE	1.144	1.129	1.125	1.536	1.497	1.434



- The *Informer* can handle longer inputs than vanilla Transformer (*Informer*$^+$).
- The *Informer* achieves better results with longer inputs under same prediction windows.

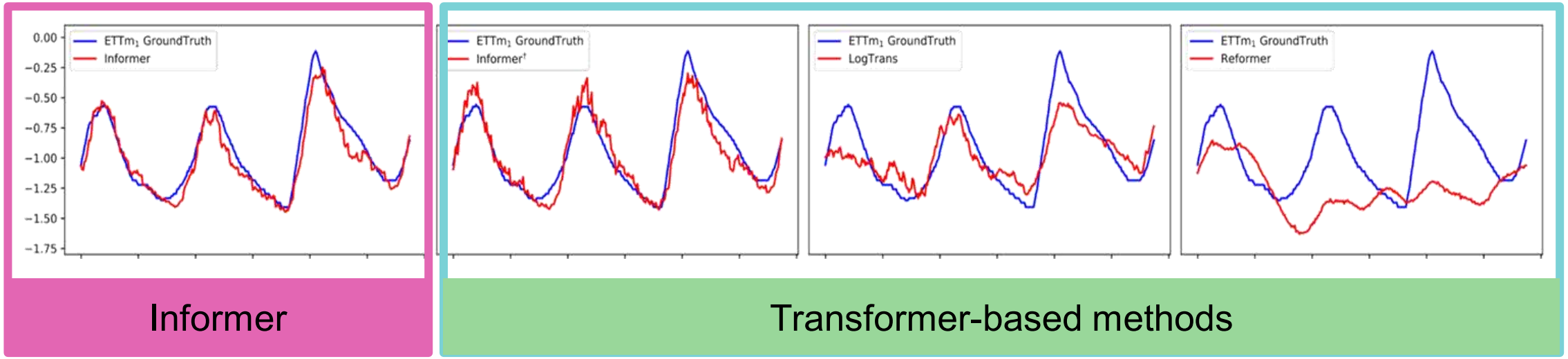
(3) Ablation study: Generative Style Decoder

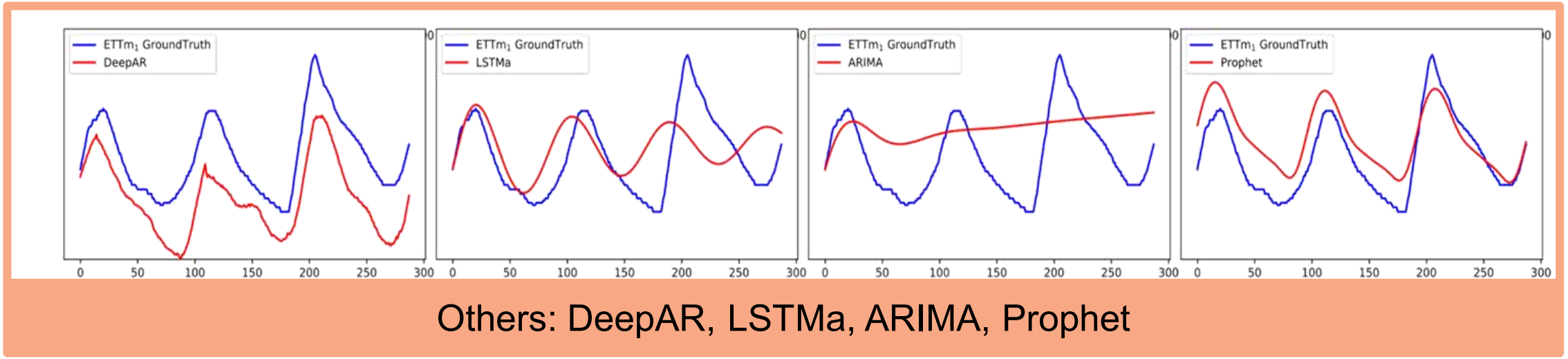
Prediction length		336					480				
Prediction offset		+0	+12	+24	+48	+72	+0	+48	+96	+144	+168
Informer‡	MSE	0.207	0.209	0.211	0.211	0.216	0.198	0.203	0.203	0.208	0.208
	MAE	0.385	0.387	0.391	0.393	0.397	0.390	0.392	0.393	0.401	0.403
Informer§	MSE	0.201	-	-	-	-	0.392	-	-	-	-
	MAE	0.393	-	-	-	-	0.484	-	-	-	-



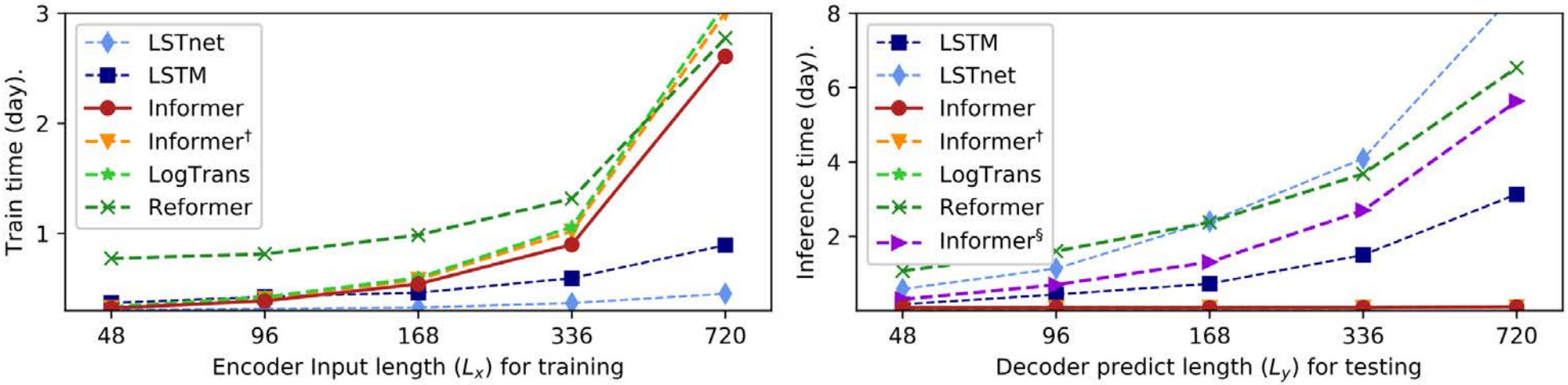
- Note that the *Informer*$^{++}$ uses the canonical self-attention and remove the distilling.
- We replace our proposed decoder with dynamic decoding, and it get worse results.
- Moreover, the generative style decoder can accurately predict under the shifting horizon without retraining the entire model.

(4) Results Visualization





(5) Runtime



The total runtime of training/testing phase.

Table 6: L-related computation statics of each layer

Methods	Training		Testing
	Time Complexity	Memory Usage	Steps
Informer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	1
Transformer	$\mathcal{O}(L^2)$	$\mathcal{O}(L^2)$	L
LogTrans	$\mathcal{O}(L \log L)$	$\mathcal{O}(L^2)$	$1^\star$
Reformer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	L
LSTM	$\mathcal{O}(L)$	$\mathcal{O}(L)$	L

[1] The LSTnet is hard to have a closed form.

The Dataset

Code: https://github.com/zhouhaoyi/Informer2020
Dataset: https://github.com/zhouhaoyi/ETDataset

Electricity Transformer Dataset (ETDataset)

In this Github repo, we provide several datasets could be used for the long sequence time-series problem. All datasets have been preprocessed and they were stored as `.csv` files. The dataset ranges from 2016/07 to 2018/07, and we will update to 2019 soon.

Dataset list

- **ETT-small**: The data of 2 Electricity Transformers at 2 stations, including load, oil temperature.
- **ETT-large**: The data of 39 Electricity Transformers at 39 stations, including load, oil temperature.
- **ETT-full**: The data of 69 Transformer station at 39 stations, including load, oil temperature, location, climate, demand.

ETT-small:

We donated two years of data, in which each data point is recorded every minute (marked by *m*), and they were from two regions of a province of China, named ETT-small-m1 and ETT-small-m2, respectively. Each dataset contains 2 year * 365 days * 24 hours * 60 mins = 1,051,200 data point. Besides, we also provide the hourly-level variants for fast development (marked by *h*), i.e. ETT-small-h1 and ETT-small-h2. Each data point consists of 8 features, including the date of the point, the predictive value "oil temperature", and 6 different types of external power load features.

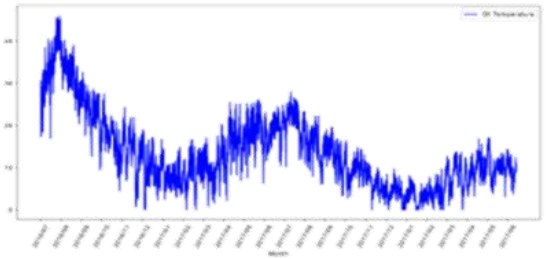


Figure 1.The overall view of "OT" in the ETT-small.

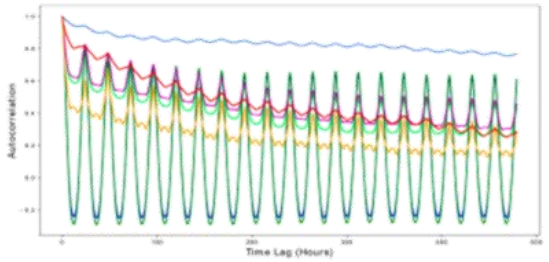
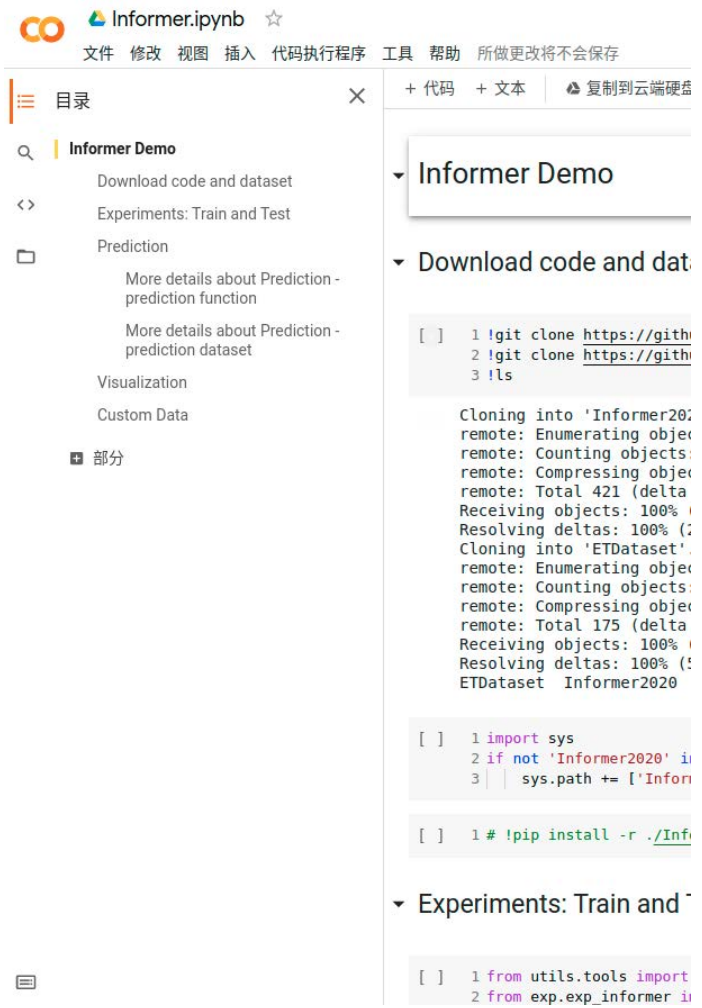


Figure 2.The autocorrelation graph of all variables.

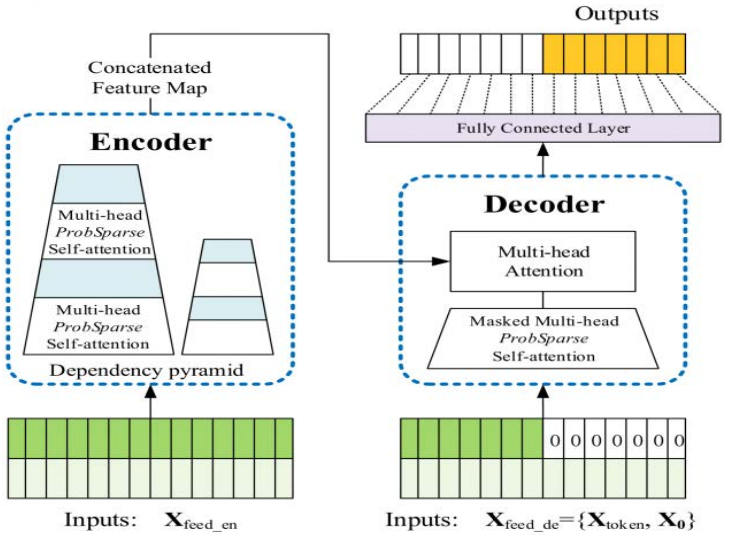


code&colab

dataset

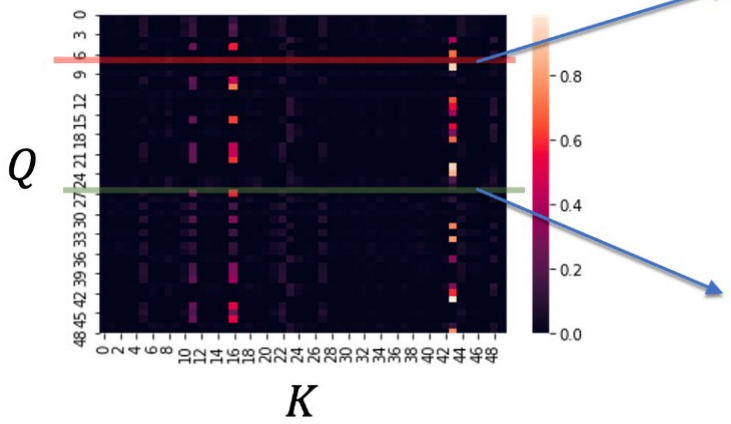
Representable: Efficient Capture of Long Dependencies

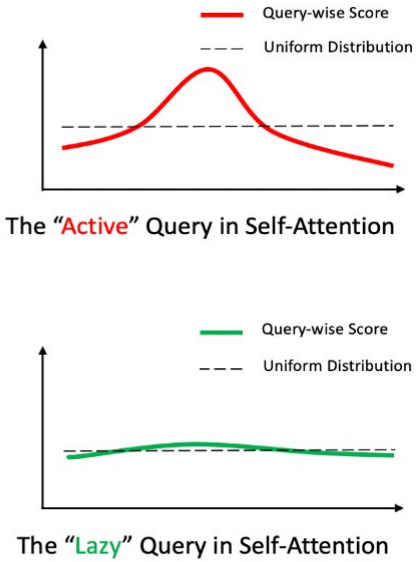
Novel Architecture: Time Series Prediction Model - Informer



Reducing Complexity: ProbSparse Self-Attention

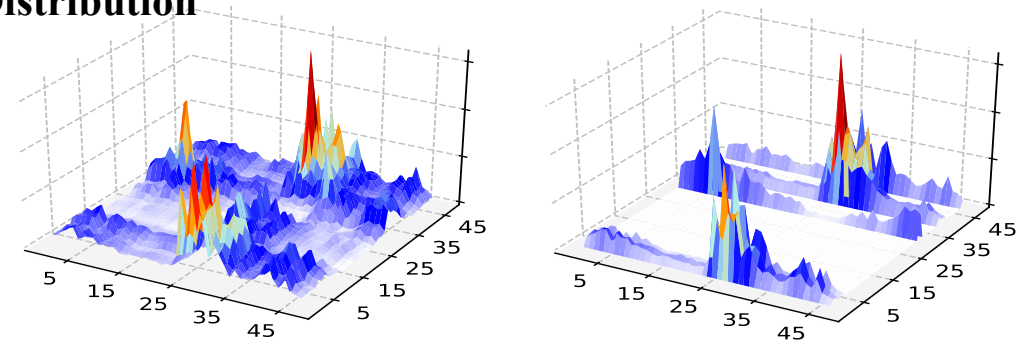
$$\mathcal{A}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{Softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V}$$

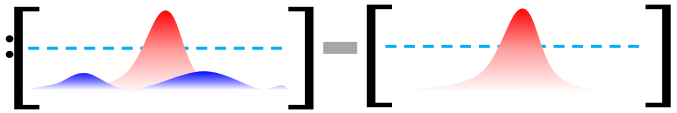




Probability Distribution-Guided Node Association Sparsity Measurement

Preserving Significant Probability Distribution



Sparse measure: 

Upper Bound Approximation: $\bar{M} = \max_j(\mathbf{q}_i\mathbf{k}_j^\top) - \mathrm{mean}_j(\mathbf{q}_i\mathbf{k}_j^\top)$

Logarithmic linear storage and computational complexity : $\mathcal{O}(L^2) \to \mathcal{O}(L\log L)$

[1] Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting, **AAAI 2021 (Best paper)**

Representable: Efficient Capture of Long Dependencies

Challenge 1 [Diverse Sequential Representations]

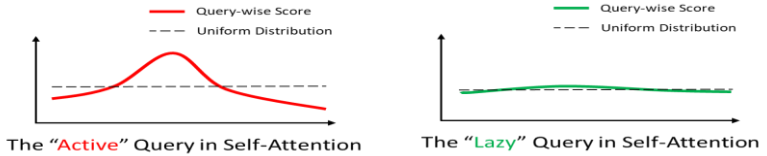
Problem: Fusion and Representation Challenges

Solution: **Unified Input Representation**

Challenge 2 [Complexity per Layer]

Problem: Quadratic Complexity of Self Attention

Solution: **Probabilistic Sparse Attention**



Challenge 3 [Long Sequence Input]

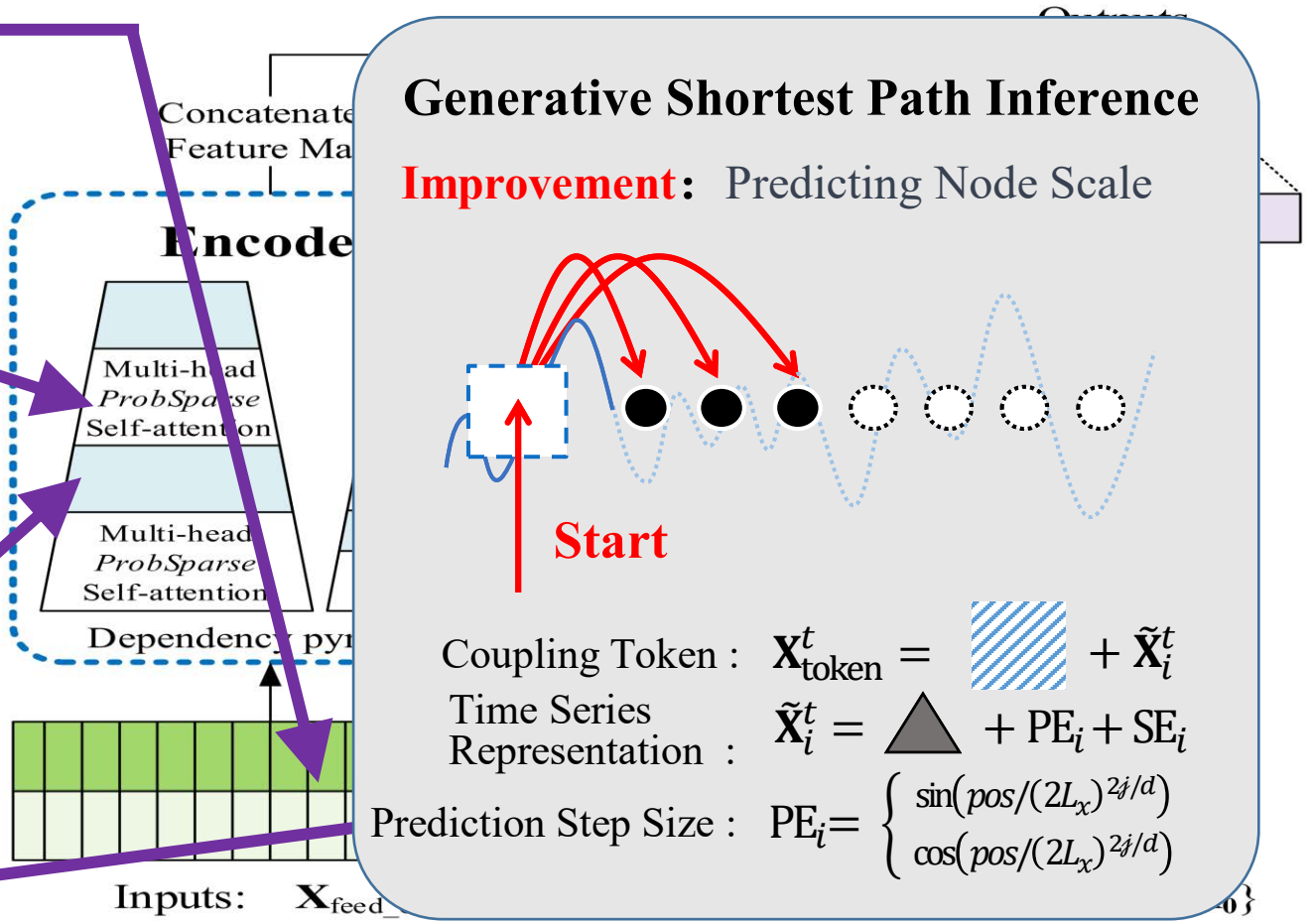
Problem: High Overhead for Multi-layer Stacking

Solution: **Attention Distillation Mechanism**

Challenge 4 [Long Sequence Output]

Problem: Accumulated Error in SA Decoding

Solution: **Generative Decoding Mechanism**



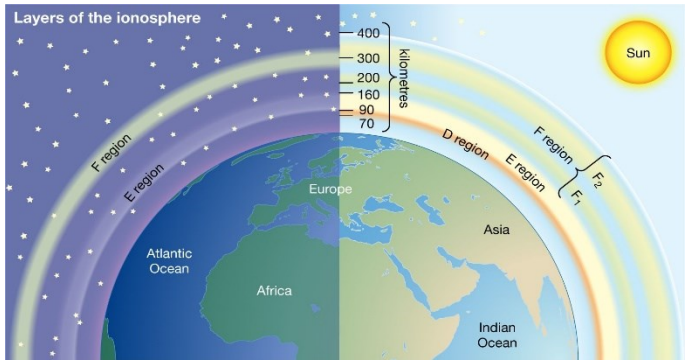
Overall Architecture of Informer

- Content -

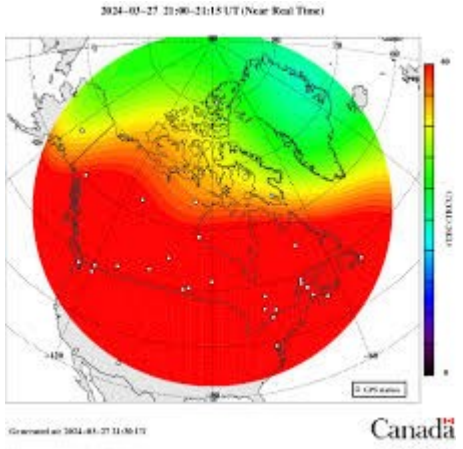
- Sequential Modeling: from Science
- The Transformer Model
- The modern architecture
- **Some applications on scientific sequences**
- Future direction

Ionformer: Global TEC Prediction

Total Electron Content

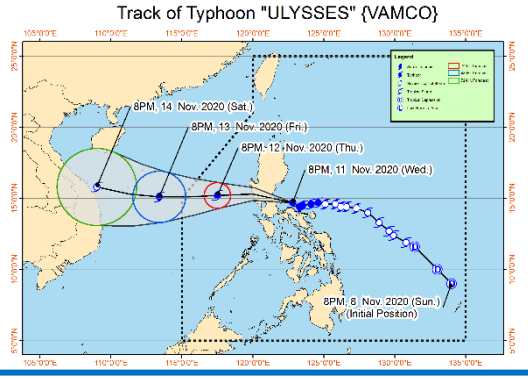


Total Electron Content (TEC) refers to the total number of free electrons along a path between two points in Earth's ionosphere, which is an important parameter for understanding the ionospheric effects on radio wave propagation.

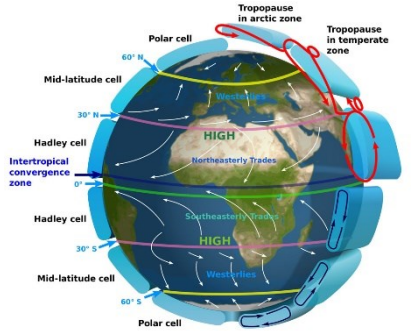


Application of TEC prediction

Radio



Meteorology



Earth Science

Challenge in TEC prediction

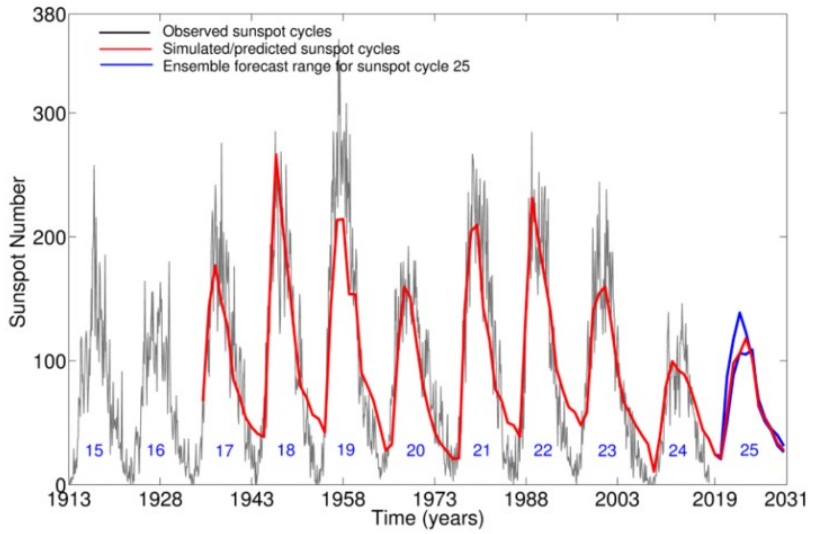
Intense ionospheric activity
Complex influencing factors

↓ **Bottleneck**

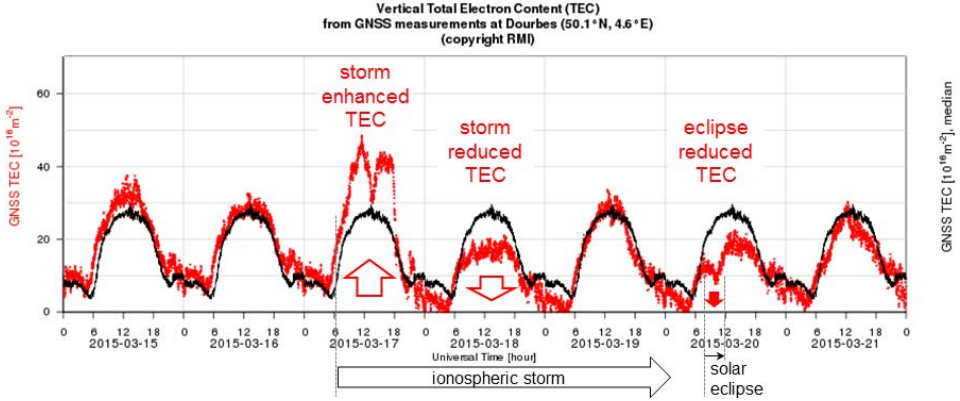
Difficulty in giving accurate prediction under **extreme climate conditions**

↓ **Reason**

Lack of model architecture tailored to the **unique properties** of the ionosphere



Solar activity cycle



Predictions

Ionformer: Sparse Attention

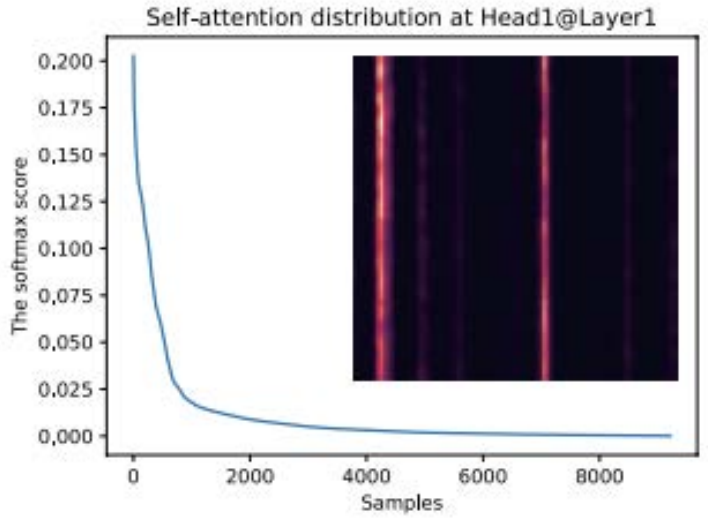
Reduce time complexity, increase robustness

Challenge

- RNN and LSTM lose information in long sequence (TEC) prediction
- Traditional attention involves quadratic time complexity and high memory complexity

Solution

- Use sparse attention mechanism to reduce time complexity
- Generate different sparse Q-K pairs for each head to avoid information loss



$$\overline{M}(q_i, K) = \max_j \left\{ \frac{q_i k_j^T}{\sqrt{d}} \right\} - \frac{1}{L_k} \sum_{j=1}^{L_K} \frac{q_i k_j^T}{\sqrt{d}}$$

Sparse Measurement | Max value | Average attention

Ionformer: Patchwise Encoding

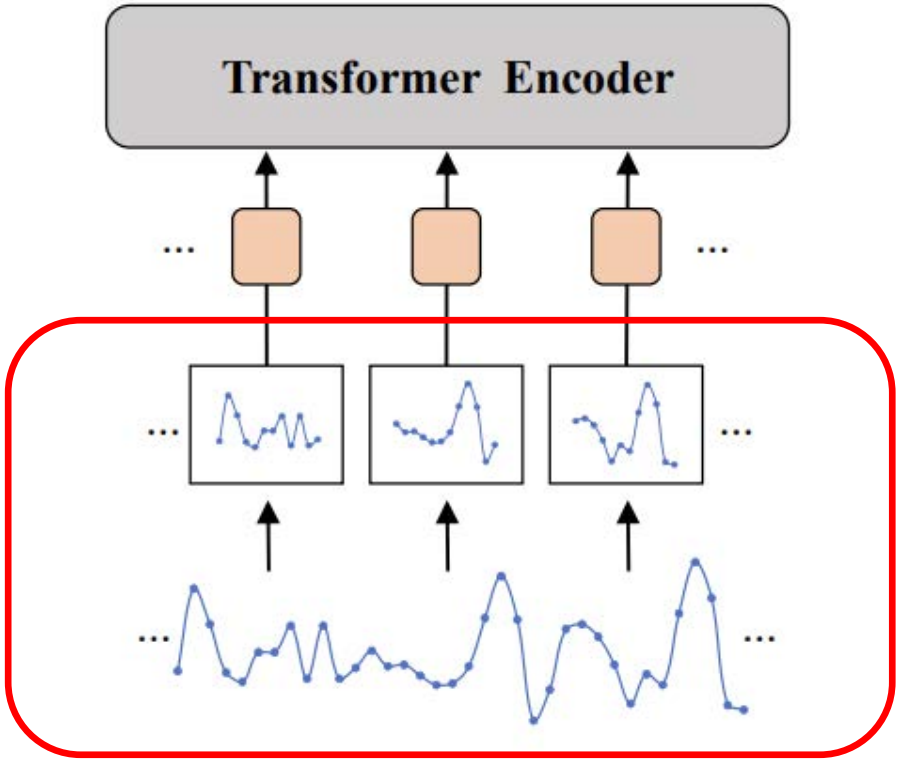
Temporal local feature aggregation

Challenge

- **Extremely long TEC sequence** limited information within single time
- Low-latitude regions are greatly **affected by solar and geomagnetic activities**

Solution

- Model information over a **period of time** rather than a single time step
- **Shift from Point-wise to Patch-wise**, reducing computational complexity

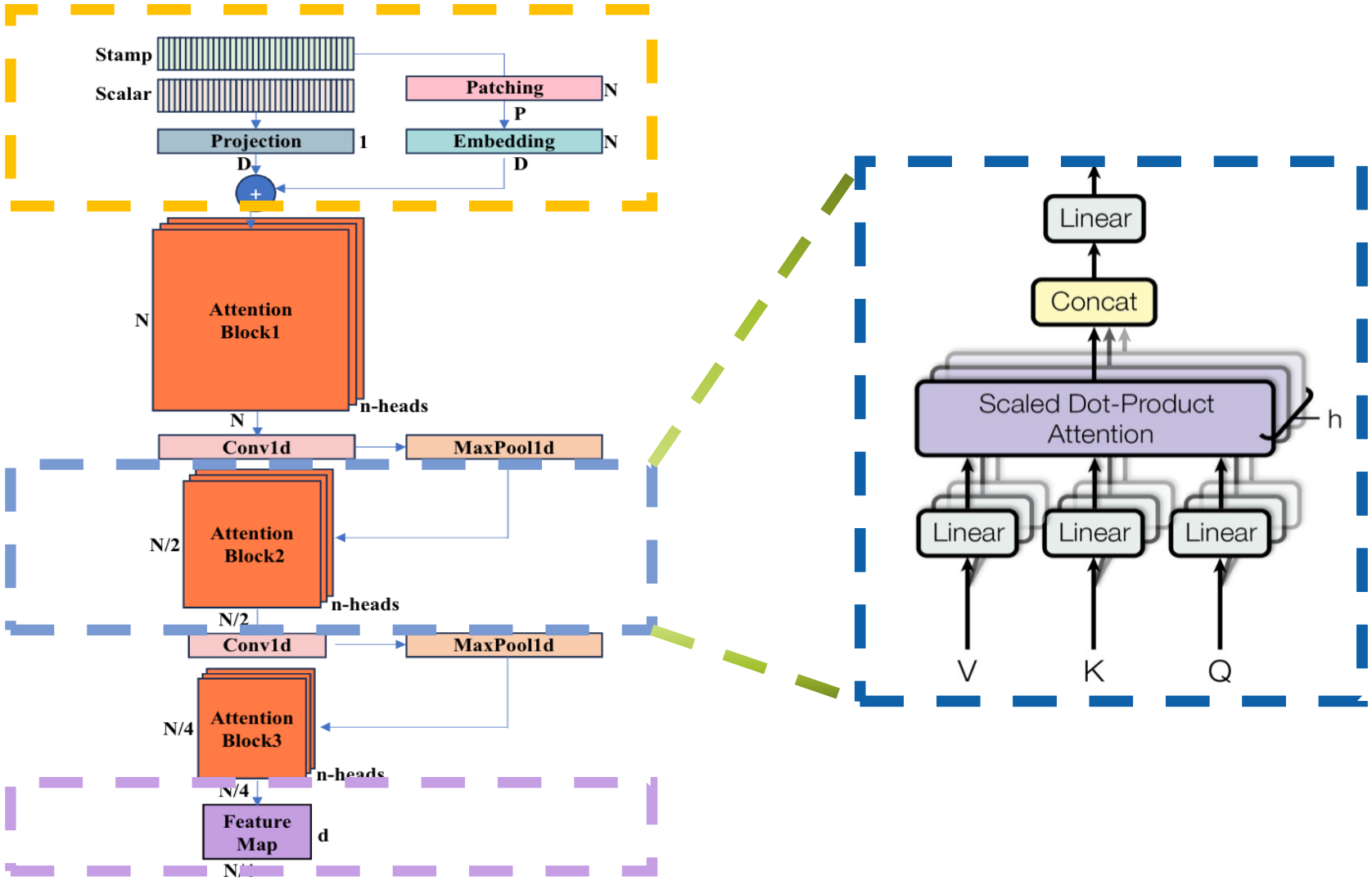


Structure of Ionformer

Time-series embeding

Sparse attention

Feature map

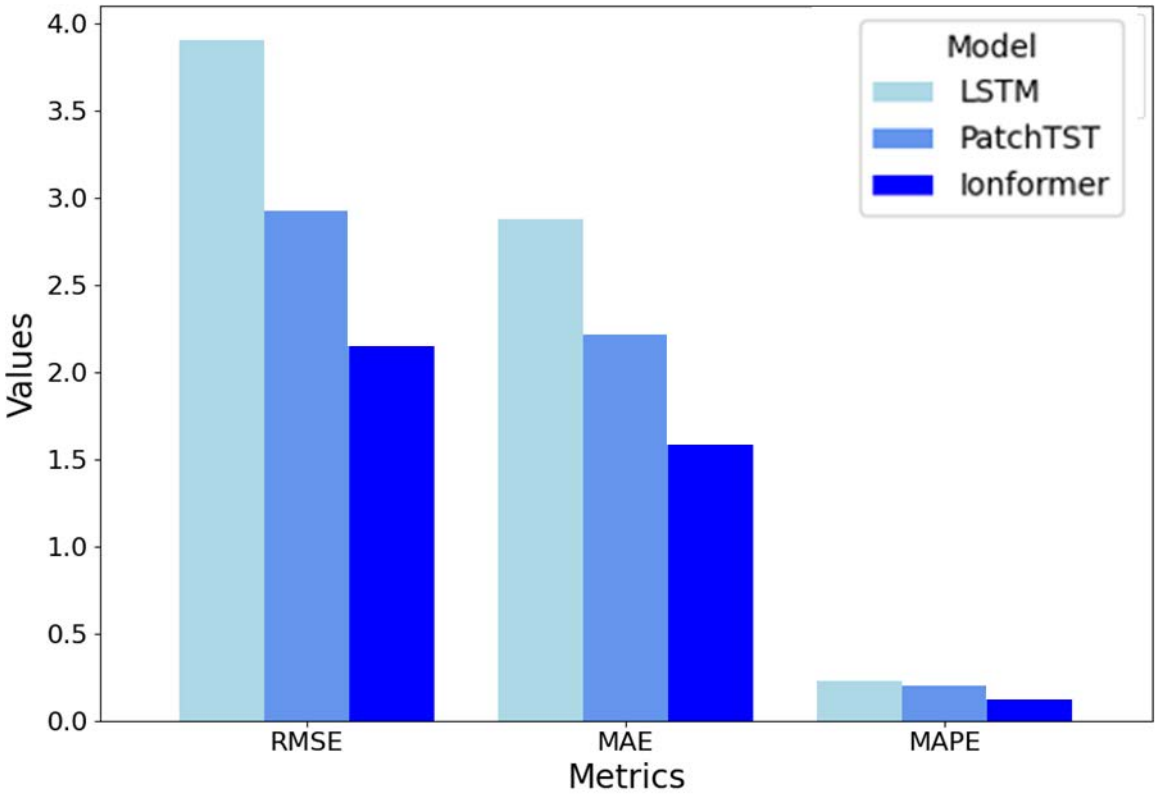


Experimental Results

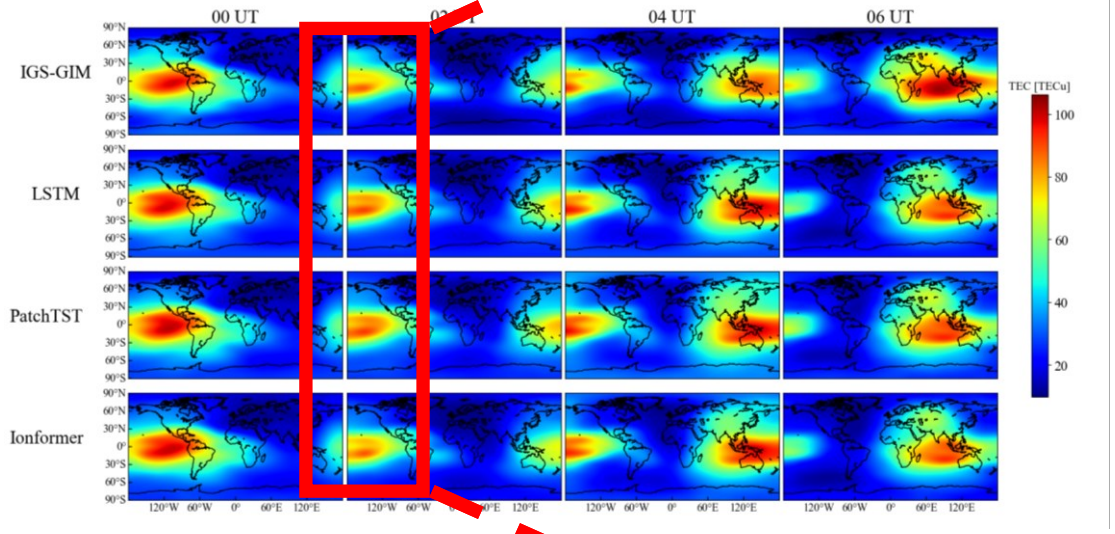
Year	2014			2017		
Metric	RMSE	MAE	MAPE	RMSE	MAE	MAPE
LSTM	4.725	3.245	0.153	1.933	1.376	0.136
Informer	4.582	3.096	0.114	1.787	1.269	0.123
PatchTST	4.322	2.910	0.135	1.739	1.221	0.119
Ionformer	**4.209**	**2.794**	**0.122**	**1.638**	**1.117**	**0.107**

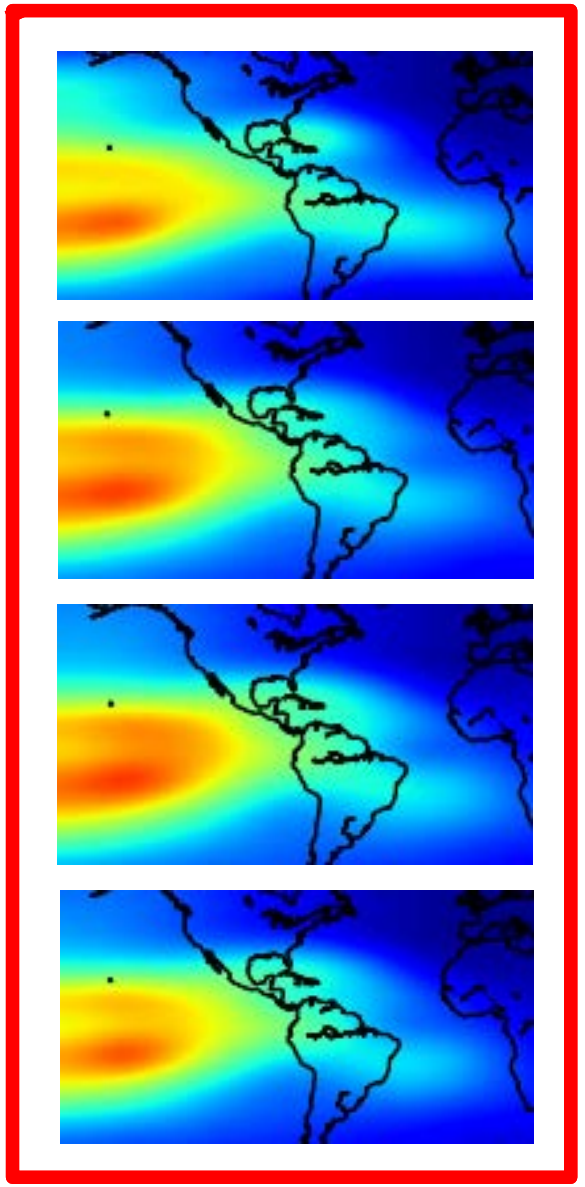
Ionformer achieves SOTA results on the TEC dataset

20% of improvement compared to baselines



Results Visualization





Ground Truth

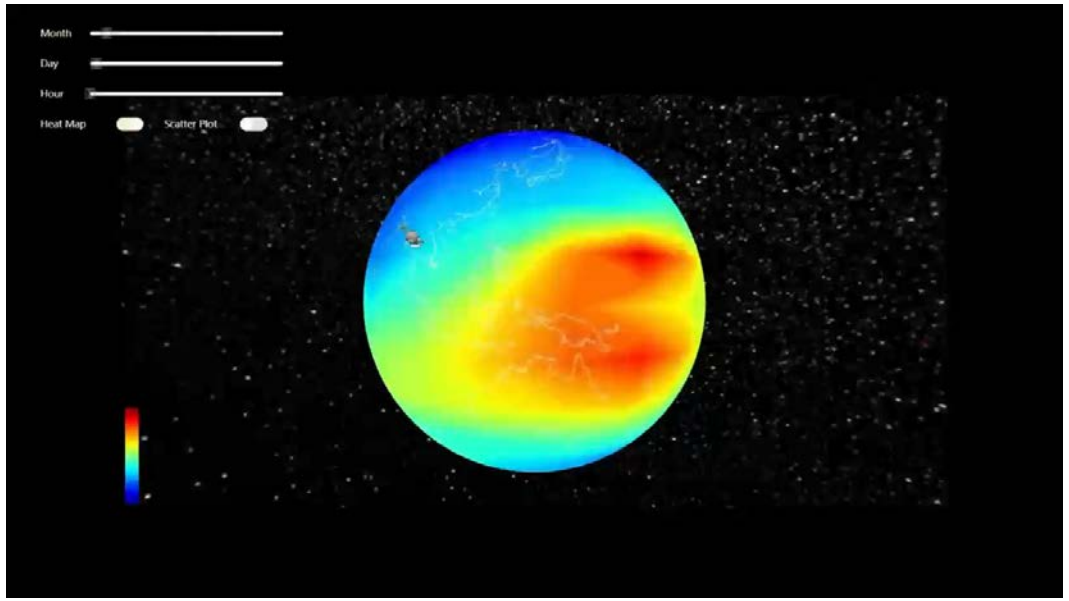
LSTM

PatchTST

Ionformer

Ionformer can better capture the local pattern of TEC

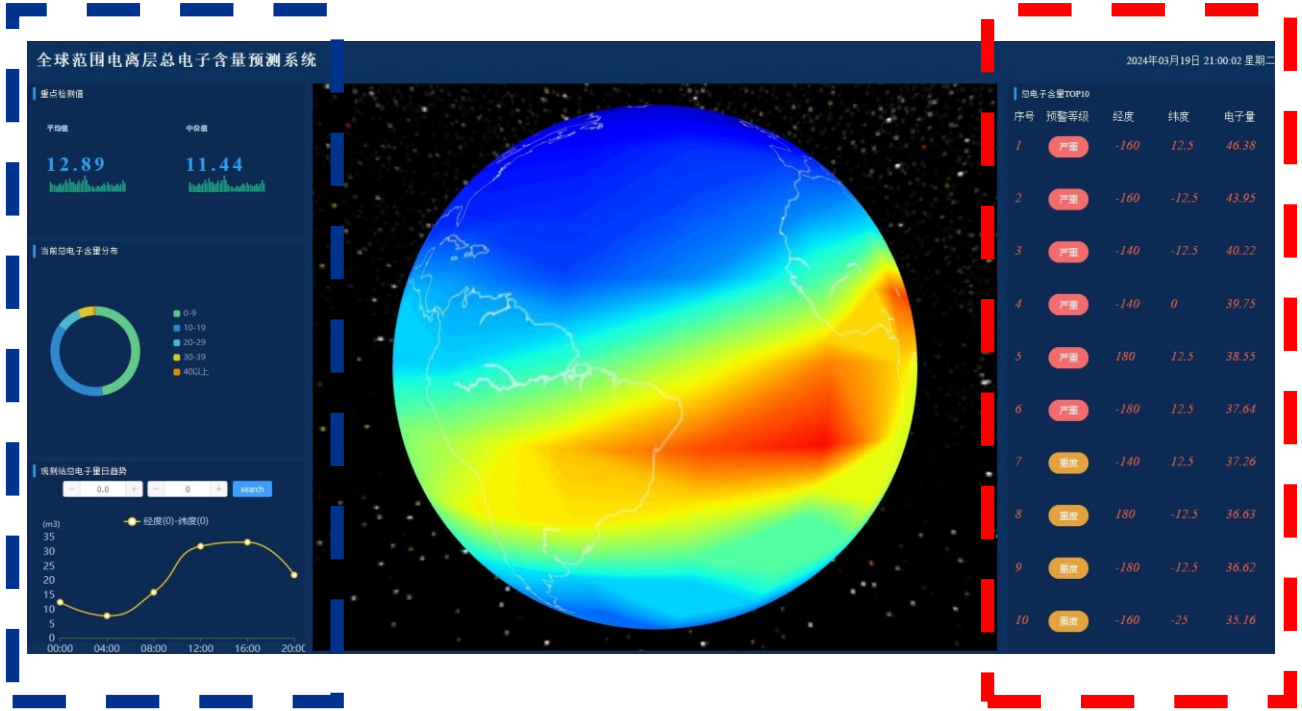
TEC Forecasting System



Prediction Results

Live Info

Anomaly Alarm



System Interface

A functional TEC prediction system based on Ionformer

学语文

- “完形填空”式训练
- 缺乏数理逻辑能力
- 涌现能力：无

学代码

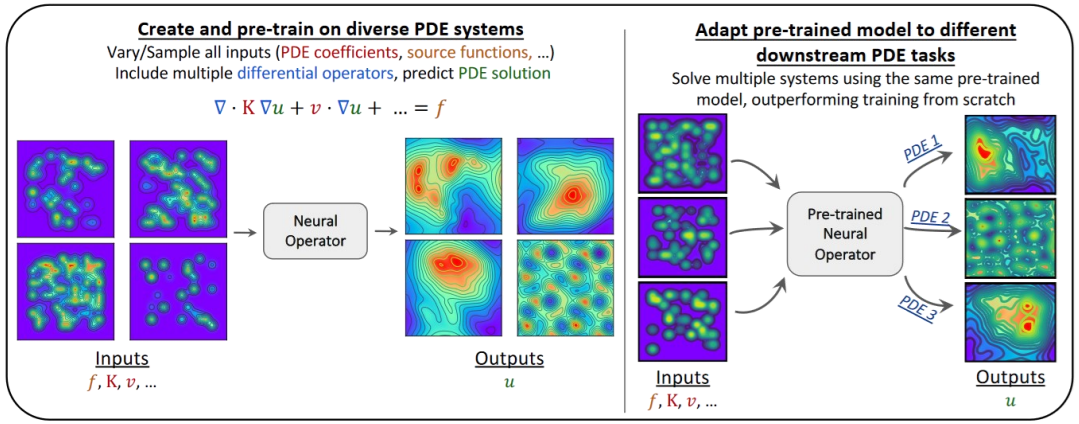
- “指令教育”式训练
- 部分任务获得逻辑推理能力
- 涌现能力：世界观、思维链、专业软件涌现能力

学物理

- “逻辑归纳”式训练
- 数据规则和物理模型的能力
- 涌现能力：科学知识生成能力、物理规律修正能力

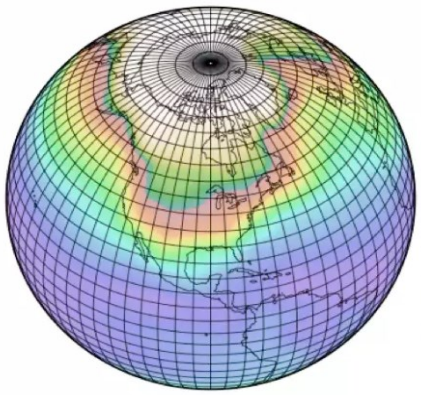
美国大规模启动科学智算大模型系列研究

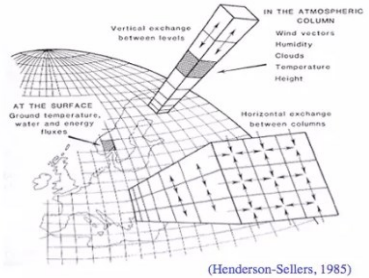
- 【新组织】2023年1月2日，由**美国能源部**牵头，布朗大学、哈佛大学等成立专门研究组织SciML
- 【新理论】2023年5月27日，**CMU**率先提出可用于高通量、高维度物理数据的新型神经算子Hyena
- 【新模型】2023年10月9日，**Meta公司**支持，图灵奖得主Yann Lecun指导发布多物理场大模型MPP
- 【新能力】2023年11月10日，**劳伦斯伯克利国家实验室**证明神经算子可扩展到2B参数，**新方程发现、物理模型修正**等能力



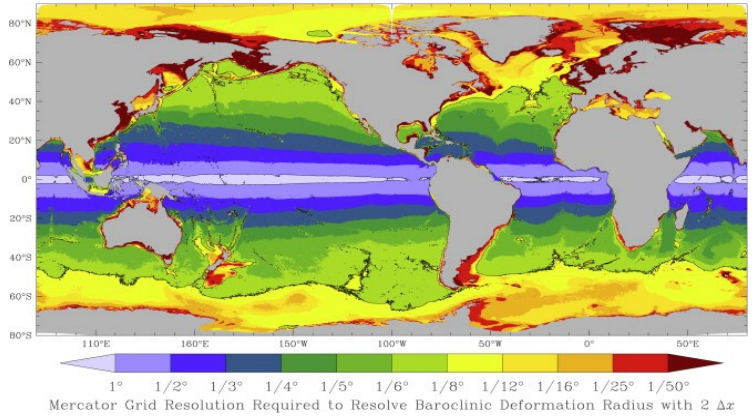
美国劳伦斯伯克利国家实验室新闻专栏报道

Building Foundation Model for Scientific Computing

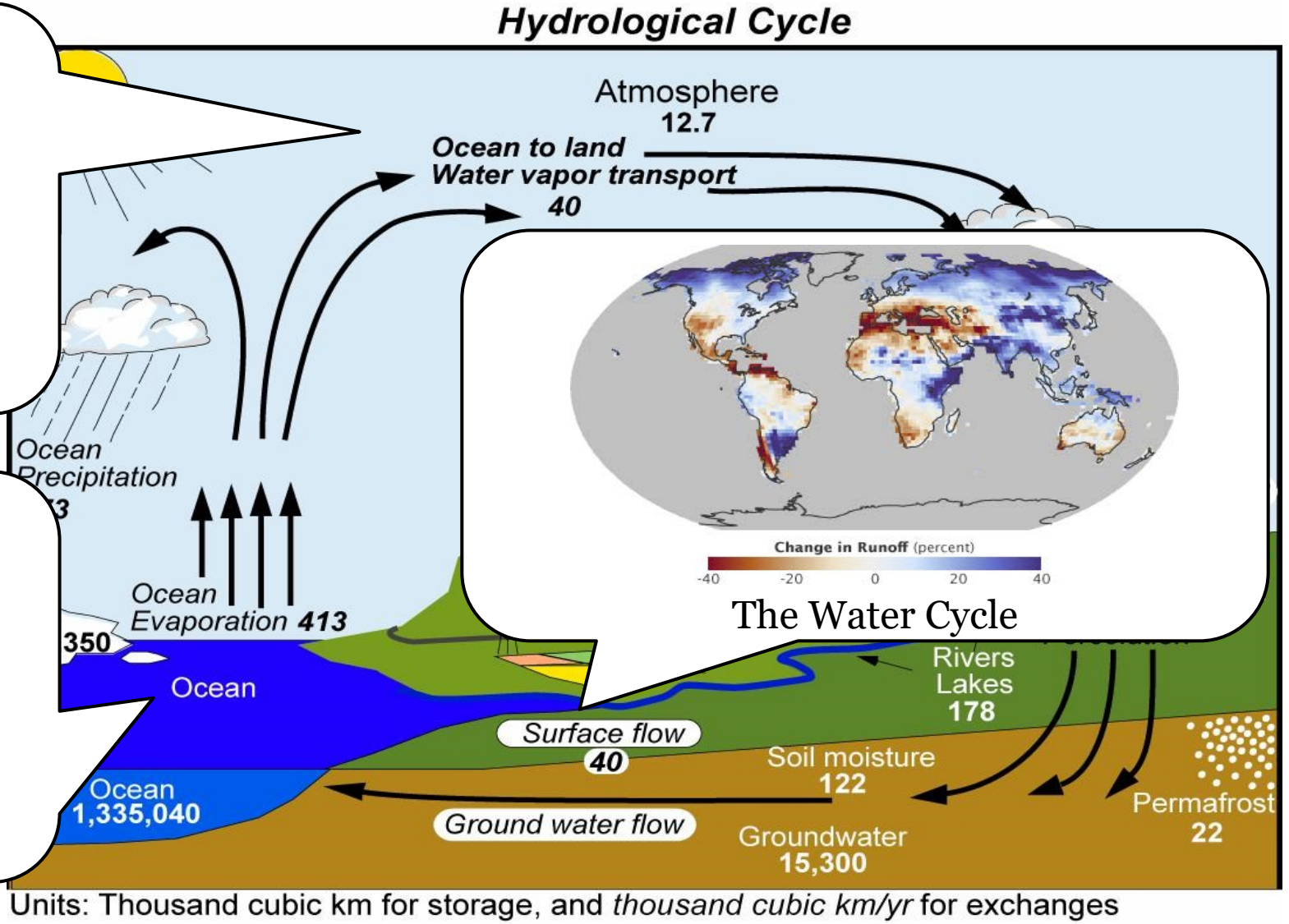




Global Climate Model



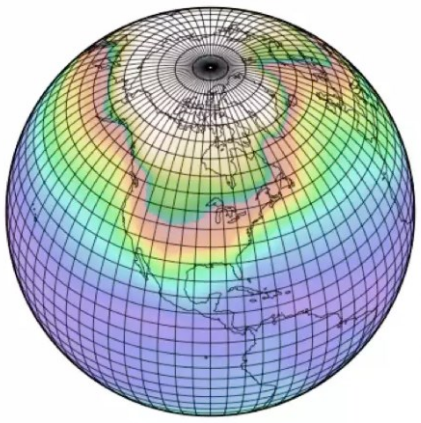
Ocean Mesoscale Eddies

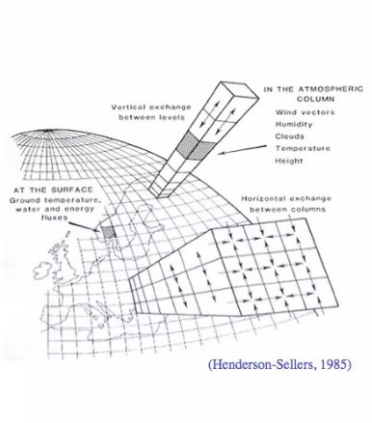


The Water Cycle

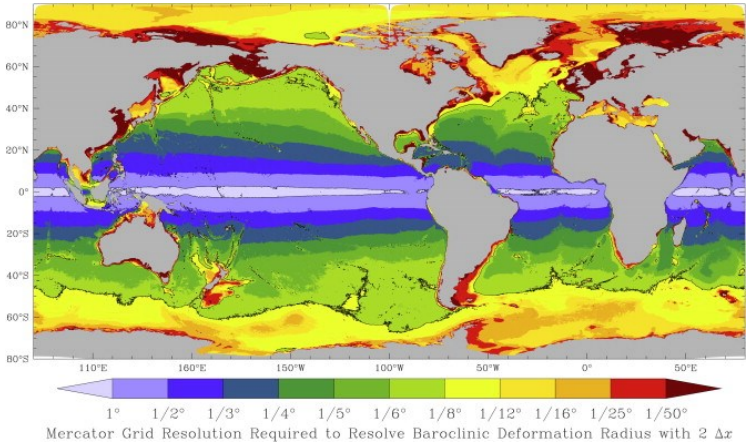
Introduction to Climate Science, Oregon State University Open Textbook, 2020

Building Foundation Model for Scientific Computing

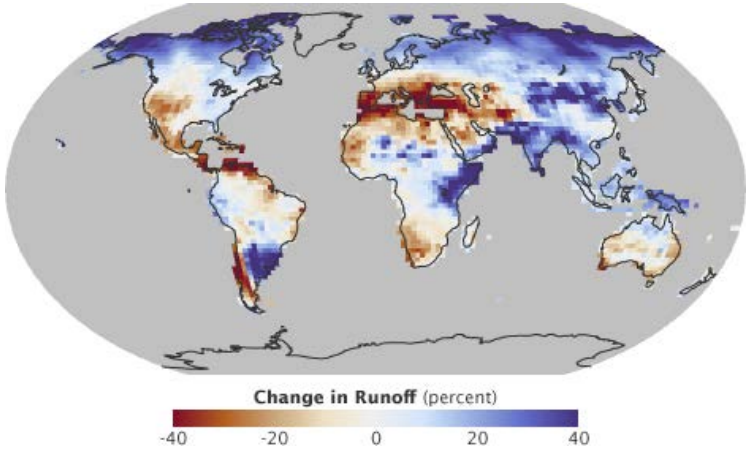




Global Climate Model



Ocean Mesoscale Eddies

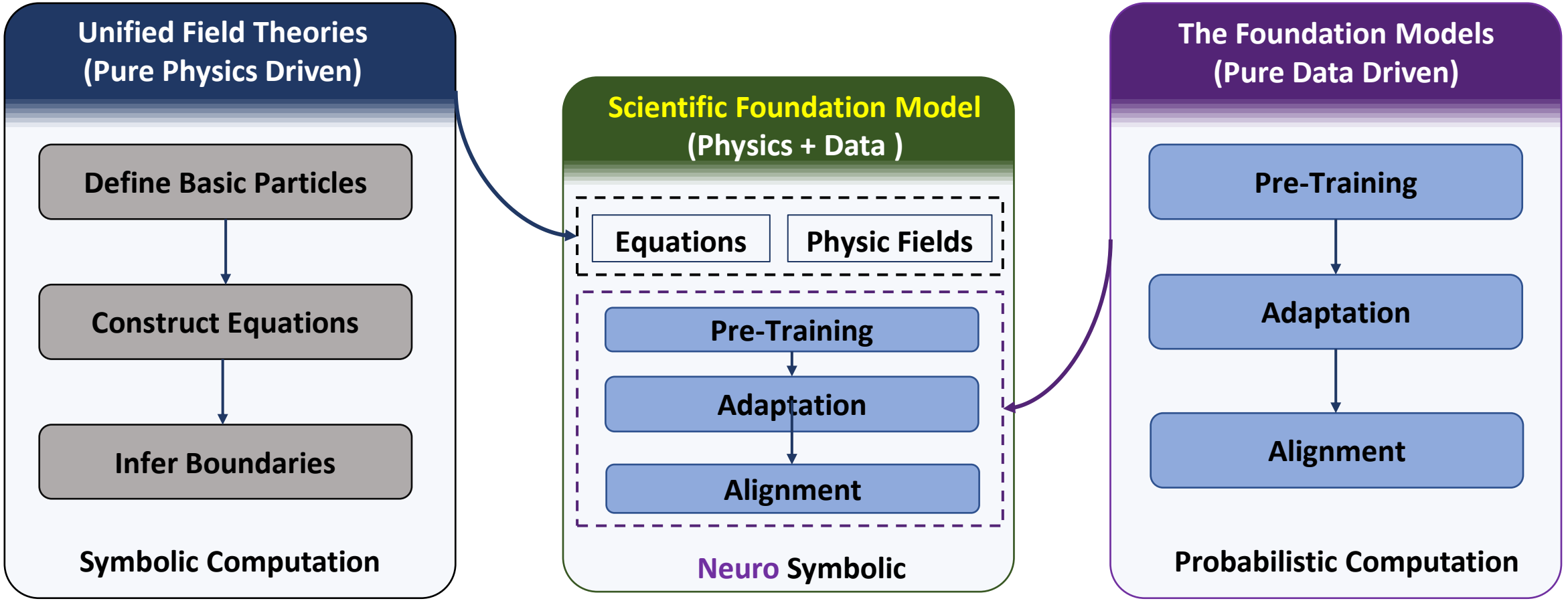


The Water Cycle

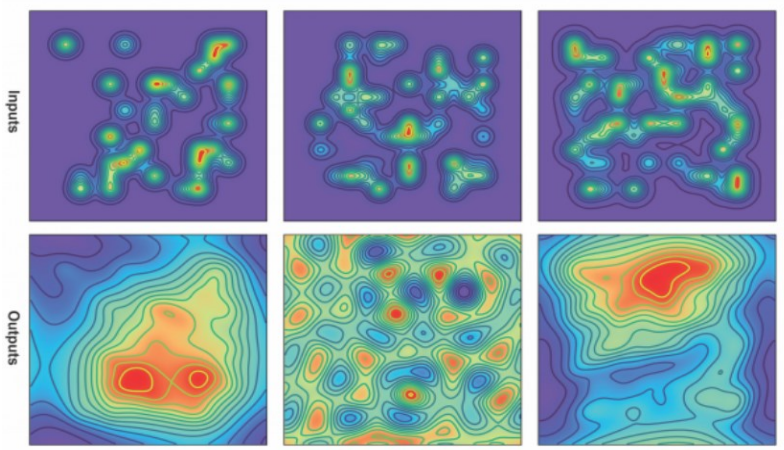
General Foundation Model?

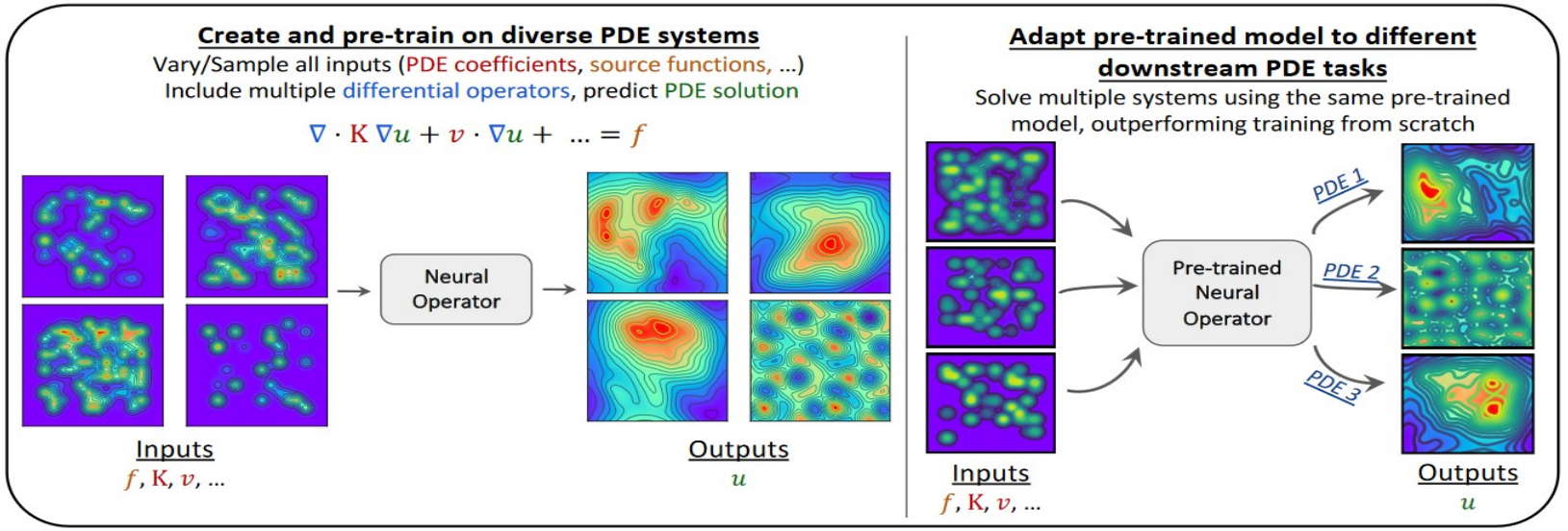
Building Foundation Model for Scientific Computing

- **Scientific Computing：** The Grant Unified Field Theories
- **Flexible Machine Learning:** The Foundation Models (GPT-4， e.t.c)
- **“Physics + data driven”:** build flexible machine learning for Scientific Computing

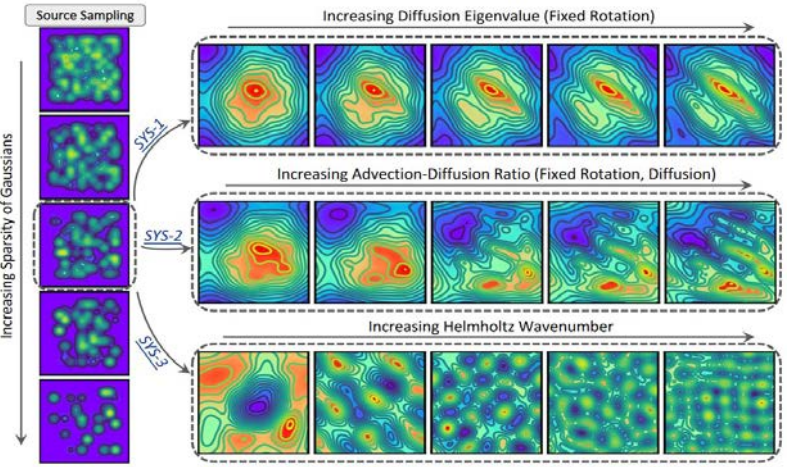


Related work

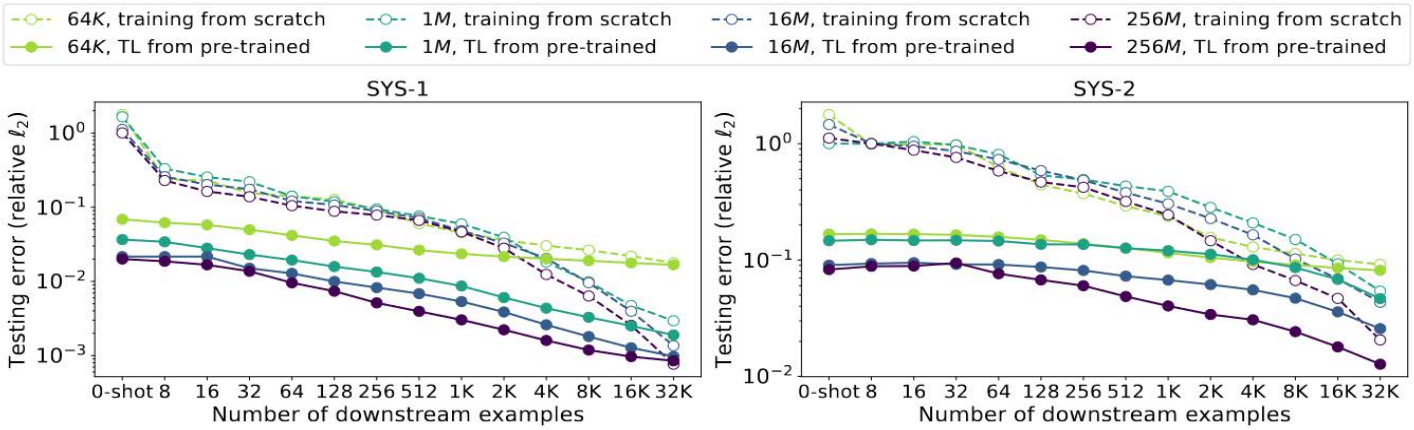




Input Function ⇨ Output Function



Pretrain + Finetune



Subramanian S, Harrington P, Keutzer K, et al. Towards Foundation Models for Scientific Machine Learning: Characterizing Scaling and Transfer Behavior[J]. arXiv preprint arXiv:2306.00258, 2023.

Building Foundation Model for Scientific Computing

Foundation model


CFD


Electromagnetics

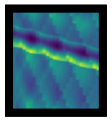

Acoustics

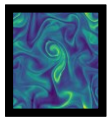

Astronomy

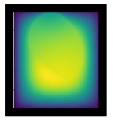
Observation

Modeling

Physical Field


Velocity


Temperature


Pressure


Sound


Electric


Gravity

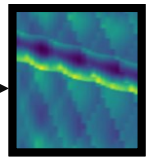
Equations

Navier-Stokes	$\rho\frac{D(\cdot)}{Dt}$
Maxwell	$\nabla\cdot\mathbf{B}\quad\nabla\times\mathbf{B}$ $\nabla\cdot\mathbf{E}\quad\nabla\times\mathbf{E}$
Multiphase Flow	$\beta\frac{\partial u}{\partial v}+\eta u$

General Scientific tasks

① Eq→Field: Forward Problem

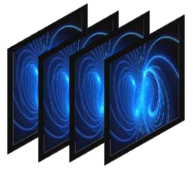
Navier-Stokes $\rho\frac{D(\cdot)}{Dt}$



--→ ?

T T+1

② Field→Eq: Inverse Problem



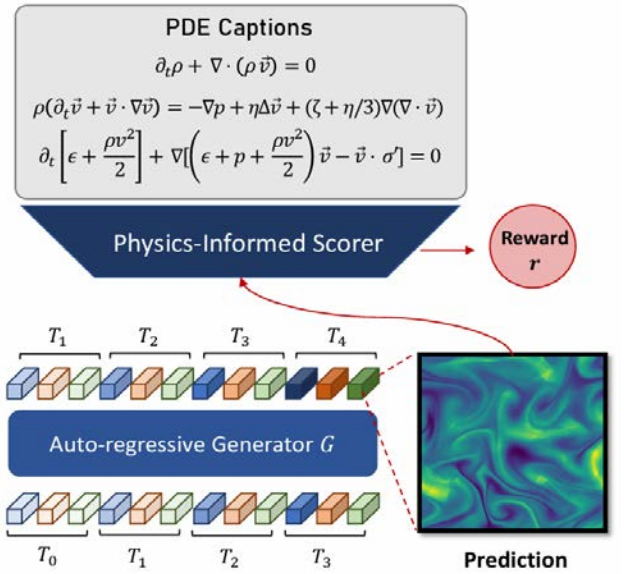
---→ PDE?

Observed Field

OmniArch: Physics-informed RL

Physics-Informed Reinforcement Learning （PIRL）

- 目标：将PDE的文本描述与对应物理系统的观测对齐
- 方法：利用对比学习，构建PDE文本描述-物理场的“CLIP”模型
- 难点：PDE文本描述（<0.1K）数据量与物理场观测点（>1M）极度不匹配



PDE Captions Augmentation （PDECap Aug）

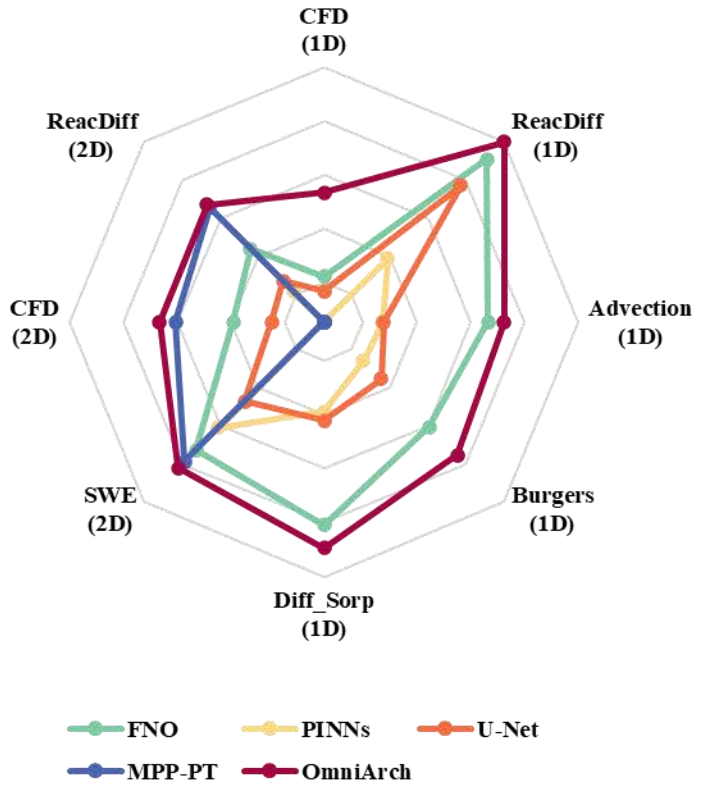
- 目标：在不改变方程语义的情况下，尽可能增加数据多样性
- 方法：规则 + 生成 + GPT-4 Check
- 效果：在PDE方程的基础上，合成约100K不同的PDE方程描述

步骤	方法名	具体操作
1	方程重写（Equation Rewriting）	运用数学恒等式，对方程进行等价变换、
2	形式转换（Form Transformation）	在微分形式和积分形式之间转换方程表达，应用格林函数等技术拓展方程的表示形式。
3	线性组合（Linear Combination）	通过线性组合导出新的方程变体
4	符号替换（Symbol Substitution）	统地将变量符号替换为其他等价符号(如将x替换为ξ),确保符号一致性和无歧义性。
5	专家物理检查（Physical Checking）	由GPT-4驱动的智能专家评估增强后的方程,过滤掉那些不符合物理定律的非法实例。

Results

Achieves the SOTA performance: (1) Previous foundation-like models like MPP-2D, PDEformer-1D often only supported a certain dimension. (2) By jointly training in 1D, 2D, and 3D, OmniArch surpasses over 40 PDE-specific models such as FNO, PINNs, U-Net with a single model.

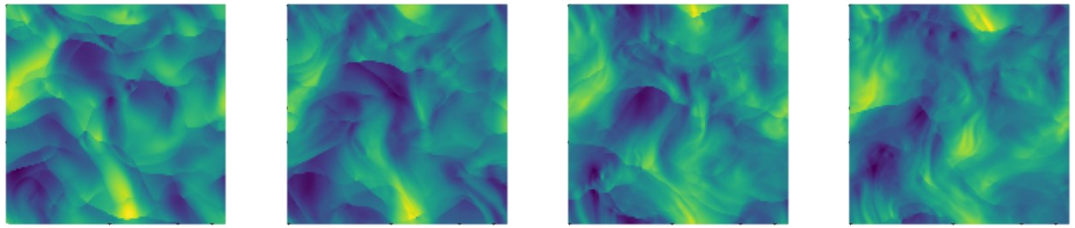
PDE		FNO	PINNs	U-Net	MPP	OmniArch	OmniArch + PIRL	相比SOTA提升
1D	CFD	1.4100	-	2.6700	-	0.0981	0.0392	97.2%
	ReacDiff	0.0005	0.2140	0.0026	-	0.0004	0.0002	64.0%
	Advection	0.0091	0.8130	0.7760	-	0.0081	0.0045	49.9%
	Burgers	0.0174	0.9450	0.3200	-	0.0067	0.0032	81.9%
	Diffusion-sorption	0.0017	0.2200	0.1500	-	0.0019	0.0006	62.2%
2D	CFD	0.2060	-	1.0700	0.0178	0.0994	0.0153	14.0%
	SWE	0.0044	0.0170	0.0830	0.0022	0.0031	0.0015	3.32%
	ReacDiff	0.1200	1.6000	0.8400	0.0098	0.0993	0.0084	14.0%
	NS-Incom	0.2574	-	1.1200	-	0.1494	0.0827	67.9%
3D	CFD	0.3050	-	0.6150	-	0.6120	0.5072	-



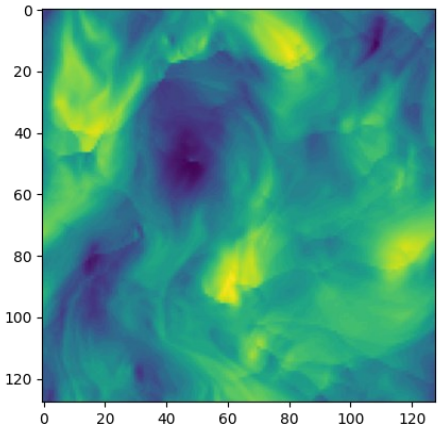
Results

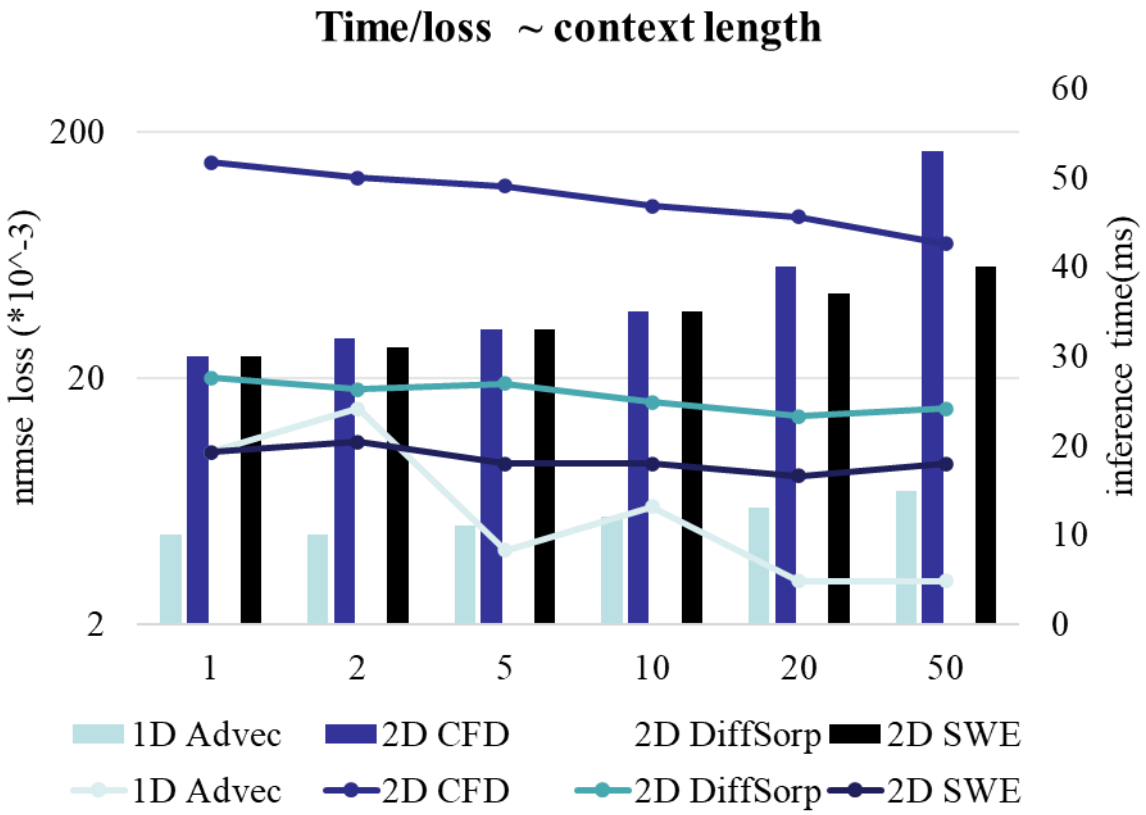
In-Context Learning PDE: **(1) OmniArch learns physics operators directly from the given pre-processed physical field variations to dynamically handle out-of-distribution (OoD) physical systems without the need for retraining. (2) In physical systems such as Advection, CFD, SWE, we observe that as the length of the input physical field sequence context increases, the solution accuracy steadily improves.**

2DCFD input context（1-4s）：



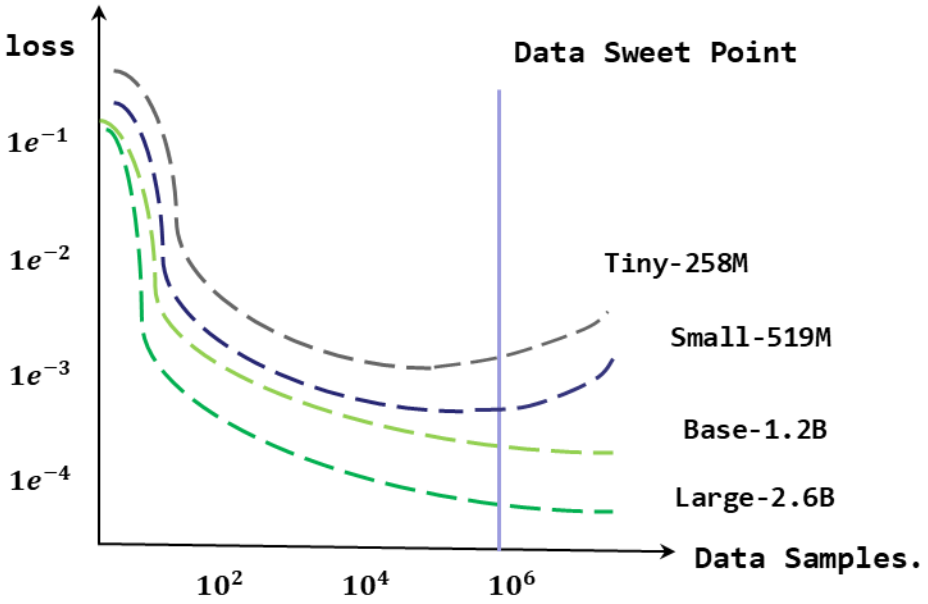
2DCFD Predictions（5-20s）：





Results

Balance of training efficiency and model size: (1) The scaling-law: From 30M to 0.5B, the model's loss function grows proportionally to the size of the parameters. When the parameters exceed 1B, the model no longer overfits. (2) Data bottleneck: The growth in model performance depends on the increase in data points. Currently, the scale of open-source data is around 10^6, limiting the maximum size of the model to about 2.6B.

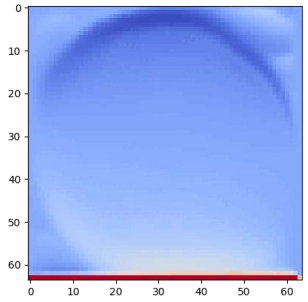


Model	Layers	Hidden	Intermediate	Heads	Params	GPU Hrs
Tiny	12	192	768	3	258M	267.6
Small	12	384	1536	6	519M	411.2
Base	12	768	3072	12	1.2B	608
Large	24	1024	4096	16	2.6B	776

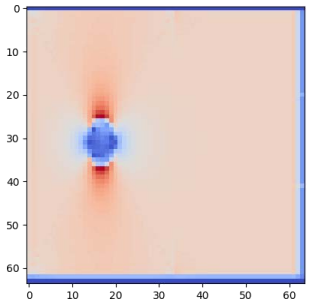
Results

Downstream tasks finetuning + few-shot testing: **(1) Fine-tuning is performed on four commonly used fluid equations: Cavity, Cylinder, Dam, and Tube. (2) Compared to models like MPP, the OmniArch model can achieve rapid improvement in accuracy with around 50 samples.**

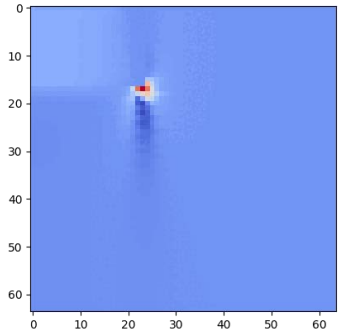
方腔流（Cavity）



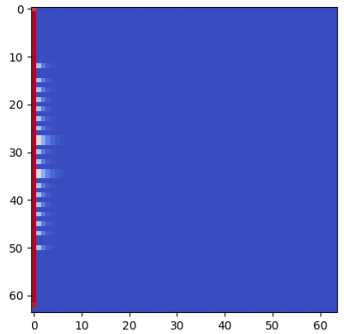
圆柱绕流（Cylinder）

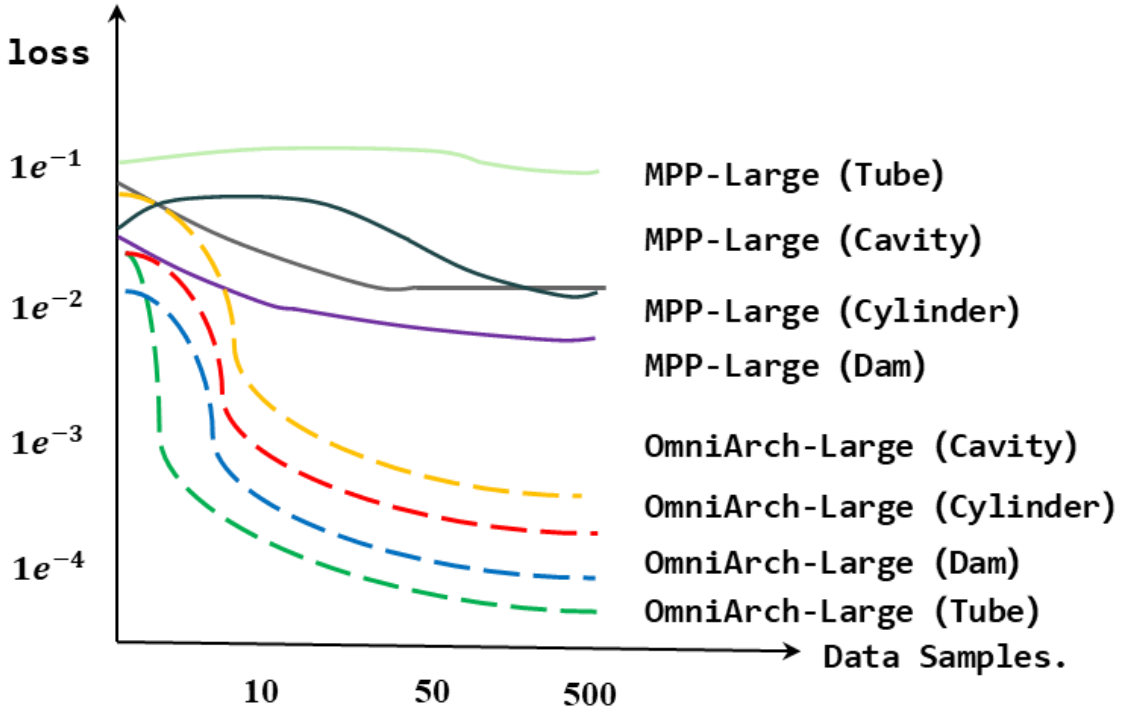


堤坝流（Dam）



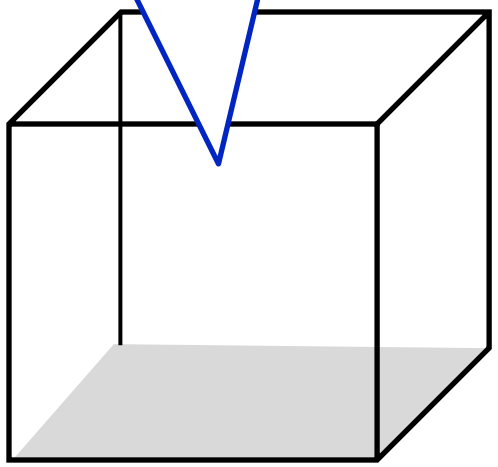
圆管流（Tube）



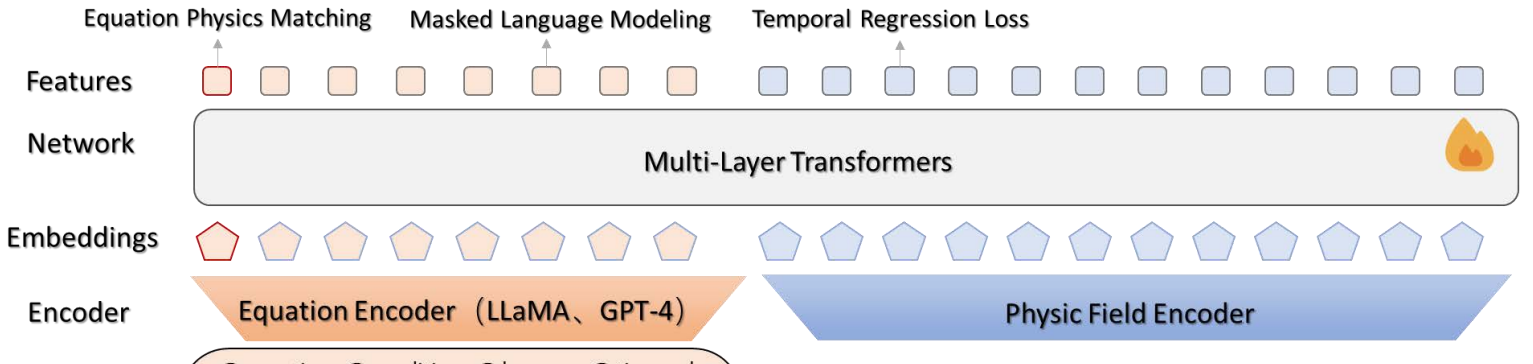


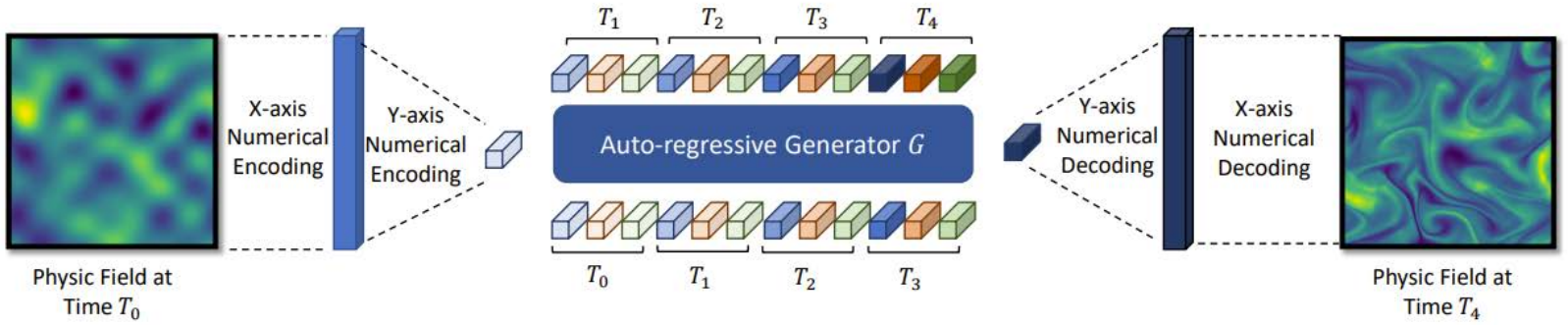
Aiming for Temporal Grounding

Temporal Knowledge



ProbSparse Attention





https://arxiv.org/pdf/2402.16014.pdf

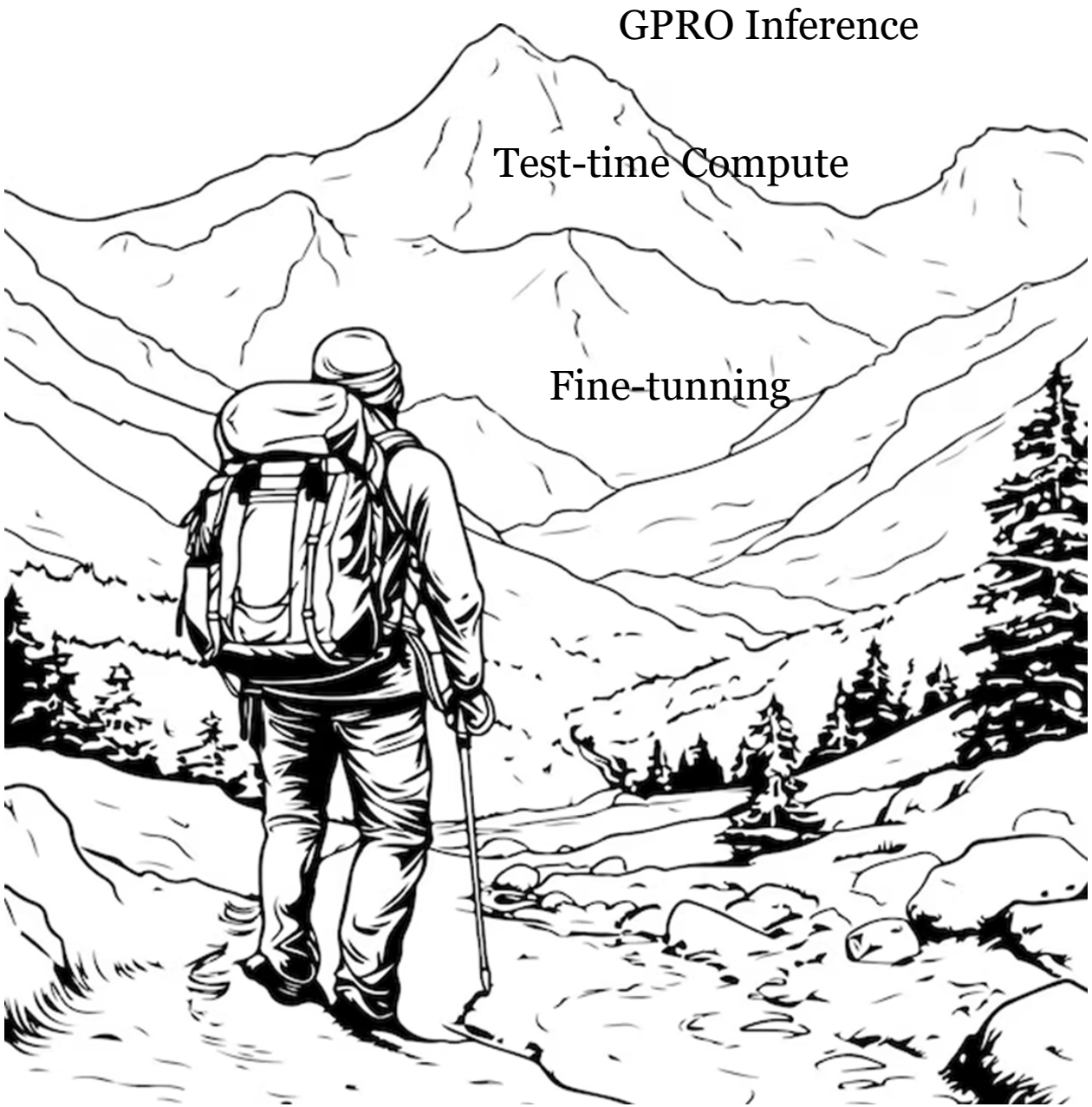
- Content -

- Sequential Modeling: from Science
- The Transformer Model
- The modern architecture
- Some applications on scientific sequences
- **Future direction**

Summary

AI4S: on the way!

Time-series: only start!



Summary

70% Data

20% Computing

10% Architecture

Unstable Trigon: What is the next sequential architecture?

Thanks!

Haoyi ZHOU, www.zhouhaoyi.com

Beihang University