LONG-RANGE ARENA

PATHFINDER TASK

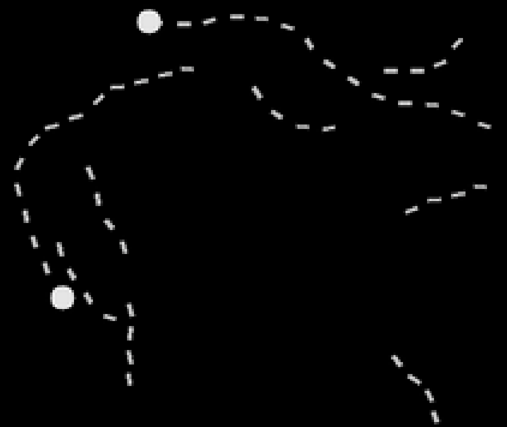
Task Explanation
Pathfinder

1

Are these points connected by a .... path ?

Task Explanation

Positive class



Negative class

Our Data:

About our Data

Meta Data Folder:

```
imgs/0 sample_0.png 0 0 1.0 6 2 2 0.5 1 1
imgs/0 sample_1.png 1 1 1.0 6 2 2 0.5 1 1
imgs/0 sample_2.png 2 1 1.0 6 2 2 0.5 1 1
imgs/0 sample_3.png 3 1 1.0 6 2 2 0.5 1 1
imgs/0 sample_4.png 4 1 1.0 6 2 2 0.5 1 1
imgs/0 sample_5.png 5 1 1.0 6 2 2 0.5 1 1
imgs/0 sample_6.png 6 1 1.0 6 2 2 0.5 1 1
imgs/0 sample_7.png 7 0 1.0 6 2 2 0.5 1 1
imgs/0 sample_8.png 8 0 1.0 6 2 2 0.5 1 1
imgs/0 sample_9.png 9 1 1.0 6 2 2 0.5 1 1
```

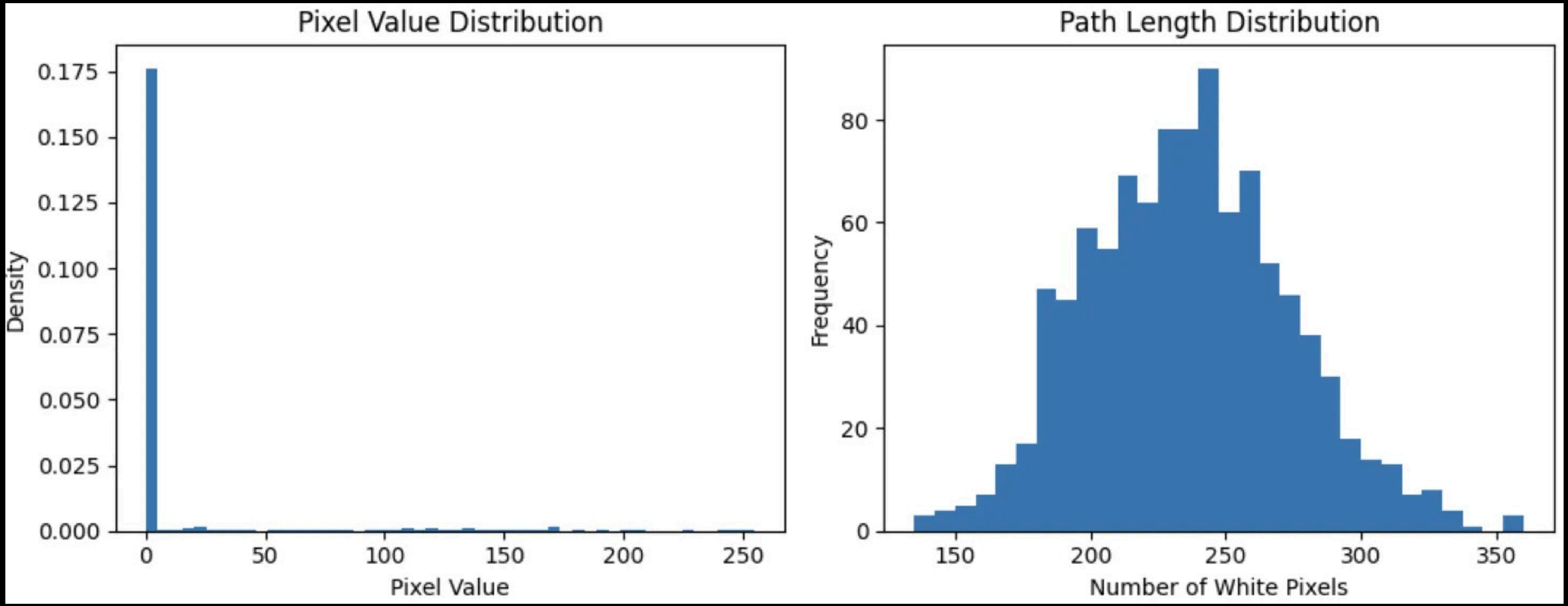
About our Data


```
Dataset Information (easy):
Total samples: 199800
Positive samples (connected): 99985 (50.04%)
Negative samples (not connected): 99815 (49.96%)
Image shape: (32, 32)
Data type: uint8
Value range: [0, 255]
```

Highly imbalanced feature space:
Background pixels (0): ~98% of image
Path pixels (255): ~2% of image

This sparsity creates a specific challenge for attention mechanisms

About our Data 2



Trained models

RNN

GRU

LSTM

TCN

Linformer

ViT

BEiT

Performer

- Structured Self Attention
- Layered

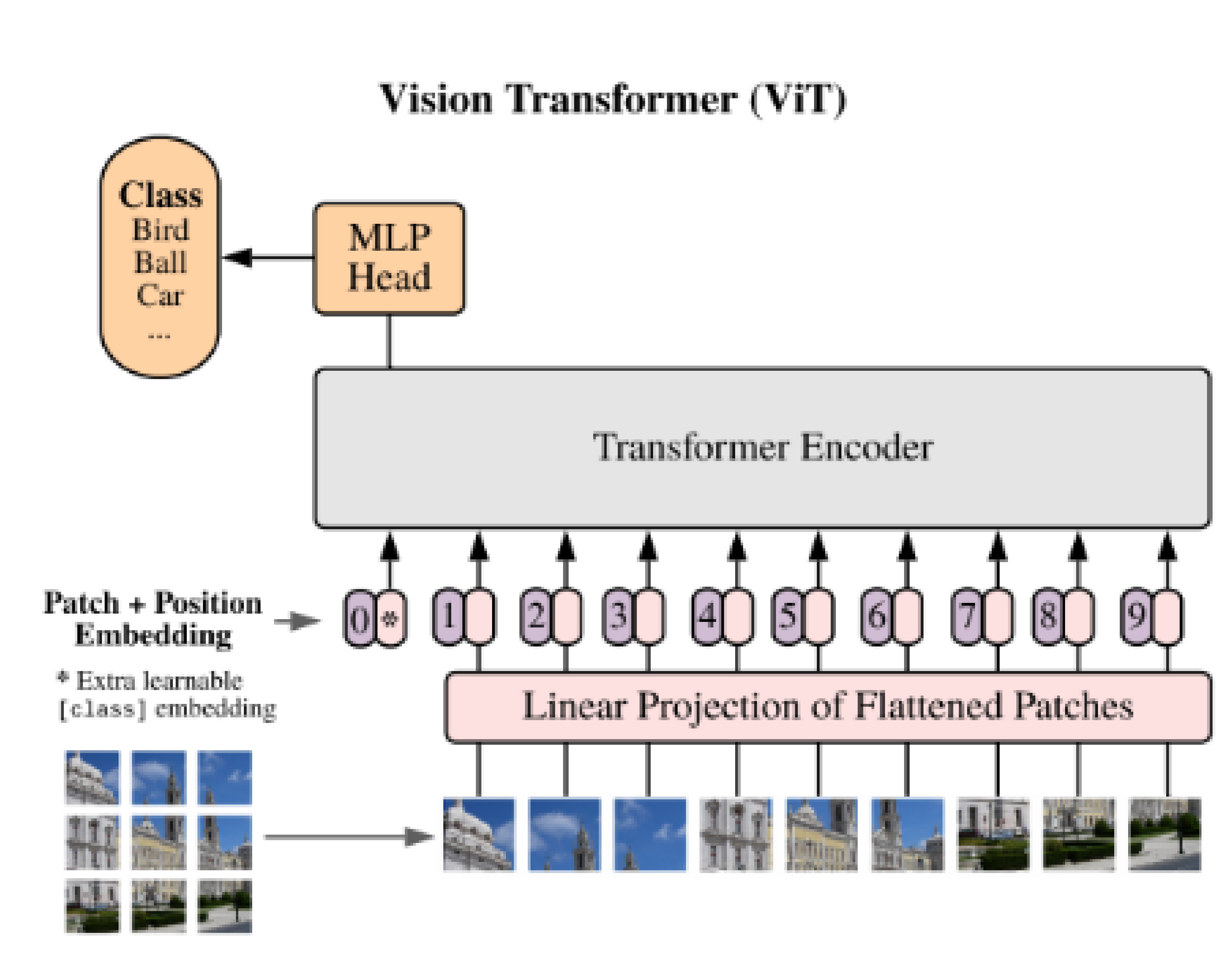
Core without improvements

Perceiver

Vanilla Transformer

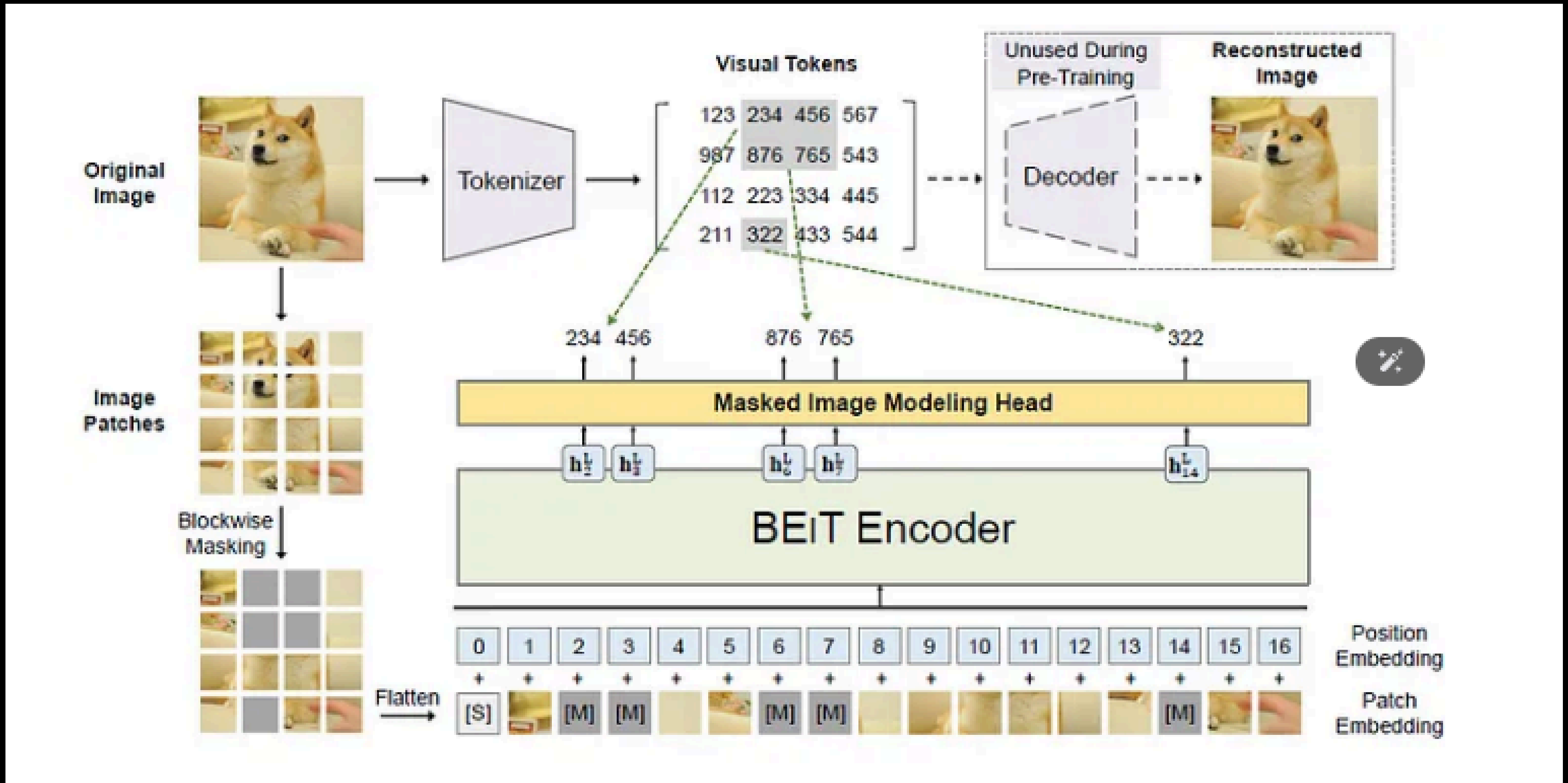
LongFormer

VIT



1. Image Patching and Embedding & Positional Encoding
2. Transformer Encoder Layer
   - Multi-Head Self-Attention (MSA)
   - Feed-Forward Network (FFN)
3. MLP Head (Classification Head)

BeiT



Overview of BEiT pre-training

Performer

Traditional Attention:

$\mathrm{softmax}(Q * K^T) * V$

Performer computes:

$\phi(Q) * \phi(K)^T * V$

Where $\phi(x)$ is a kernel function that maps inputs into a higher-dimensional feature space.

$\phi(x)$

Maps queries and keys into a new space where their dot product approximates the softmax kernel.

Performer is a transformer variant designed to reduce the computational and memory cost of self-attention.

- Uses Fast Attention via Positive Orthogonal Random Features (FAVOR+) to approximate self-attention with linear complexity $O(n)$.

Key Idea: *Self-Attention Bottleneck*

Traditional Attention Formula:

$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}((Q * K^T) / \sqrt{d_k}) * V$

- Q: Query
- K: Key
- V: Value

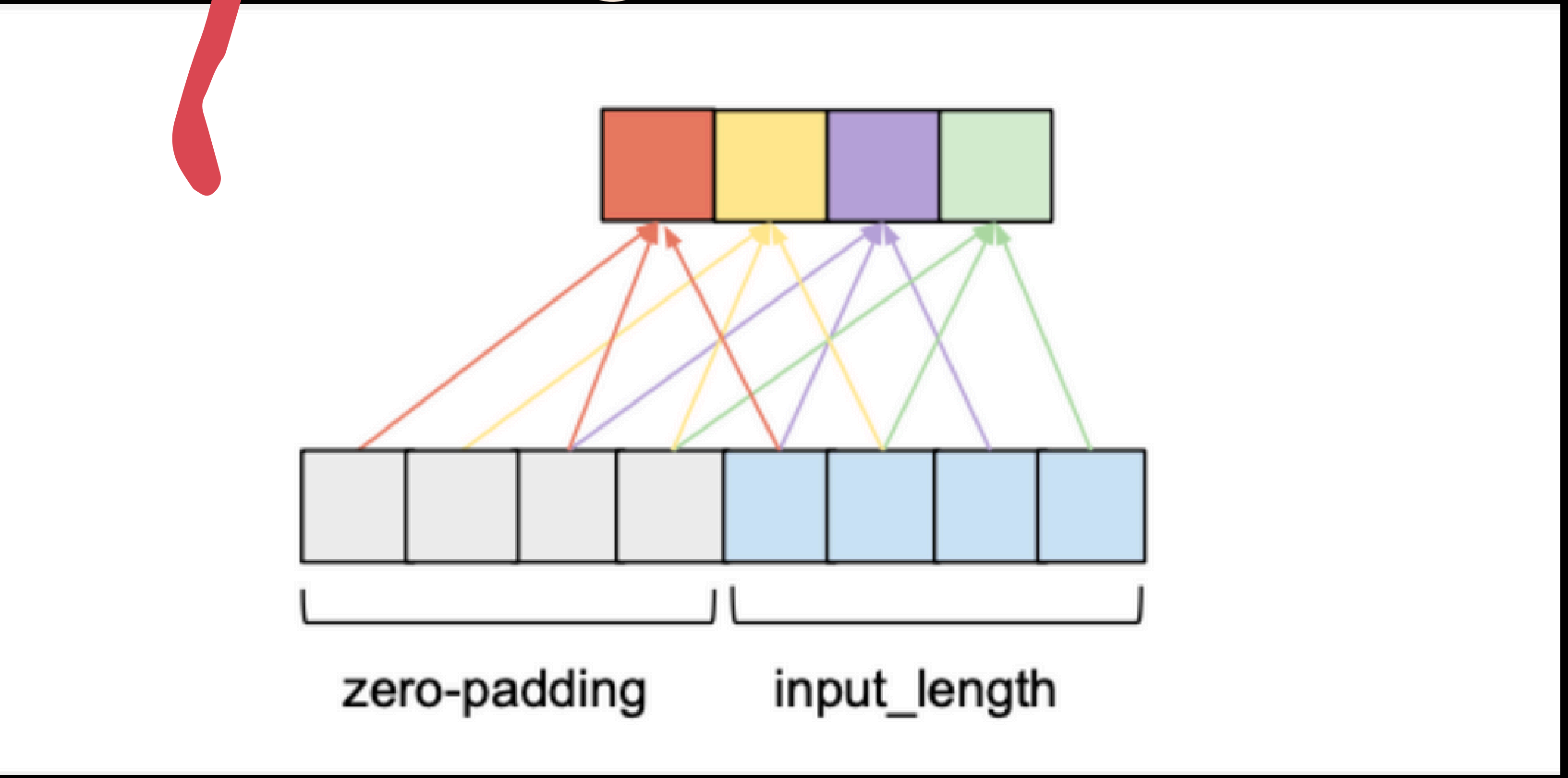
Complexity:

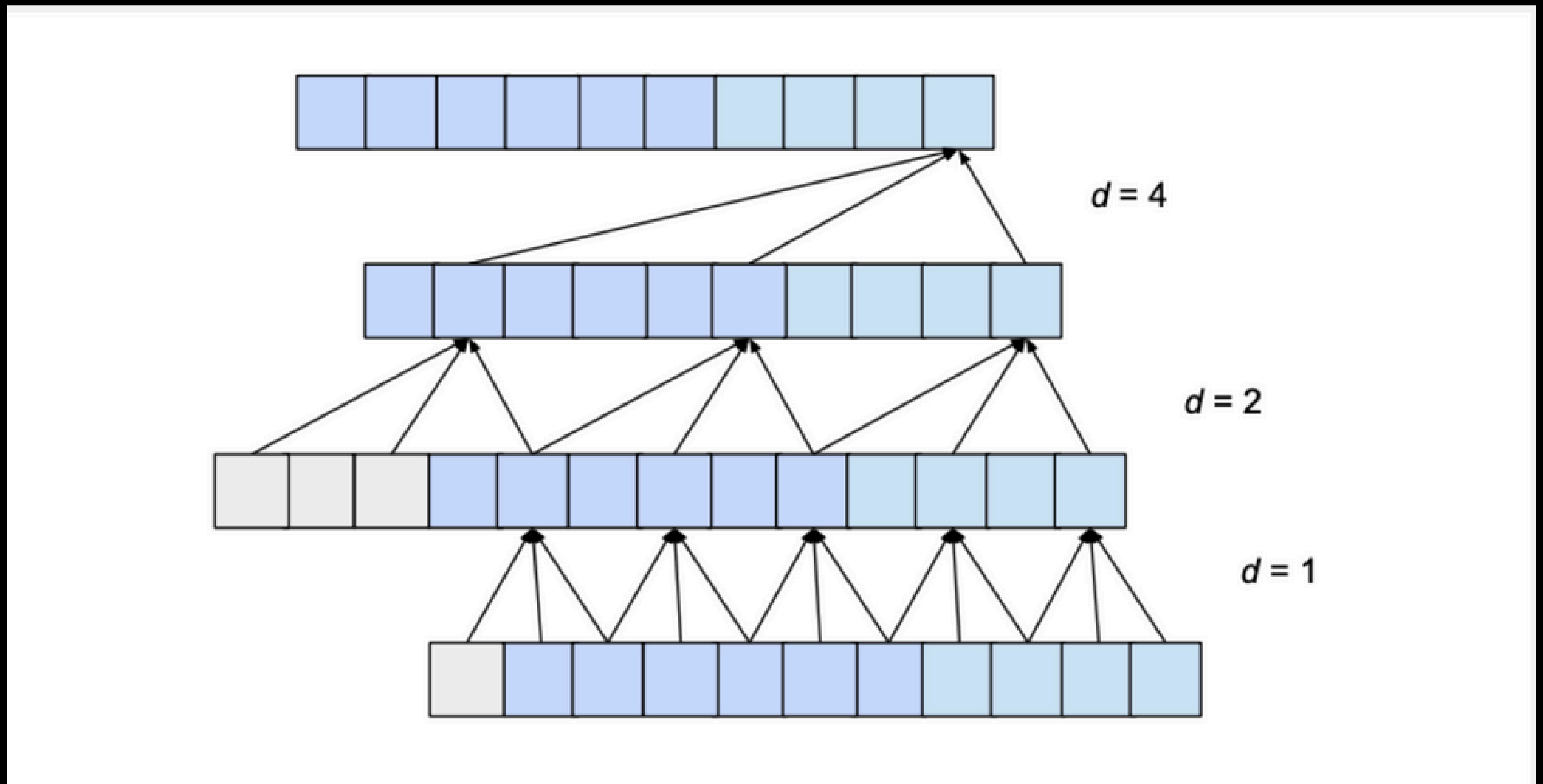
- $O(n^2)$ because it computes pairwise interactions between all tokens.

Solution by Performer:

- Replace exact attention with linear attention using kernel approximations.

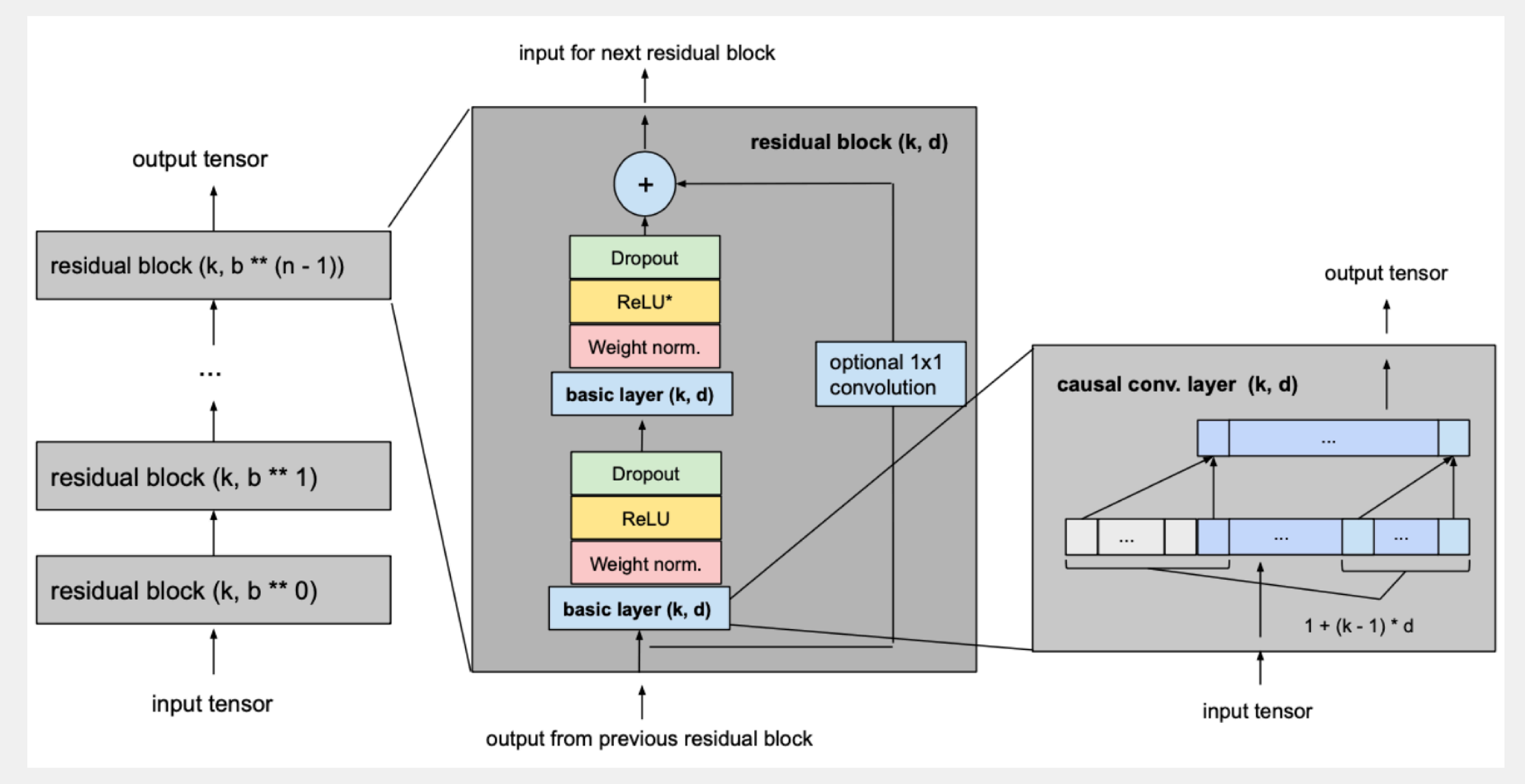
Temporal Convolutional Network





$$w = 1 + \sum_{i=0}^{n-1} (k-1) \cdot b^i = 1 + (k-1) \cdot \frac{b^n - 1}{b - 1}$$

Temporal Convolutional Network



Linformer

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W^O, \tag{1}$$

where $Q, K, V \in \mathbb{R}^{n \times d_m}$ are input embedding matrices, n is sequence length, d_m is the embedding dimension, and h is the number of heads. Each head is defined as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) = \underbrace{\text{softmax}\left[\frac{QW_i^Q(KW_i^K)^T}{\sqrt{d_k}}\right]}_{P} VW_i^V, \tag{2}$$

where $W_i^Q, W_i^K \in \mathbb{R}^{d_m \times d_k}, W_i^V \in \mathbb{R}^{d_m \times d_v}, W^O \in \mathbb{R}^{hd_v \times d_m}$ are learned matrices and d_k, d_v are the hidden dimensions of the projection subspaces. For the rest of this paper, we will not differentiate between d_k and d_v and just use d.

Linformer

Mixed Precision: Orthogonal and always used
Knowledge Distillation: Teacher Model Problem Persists

Sparse Attention: Performance Degradation with limited gained efficiency
LSH Attention: Large Constant in Complexity

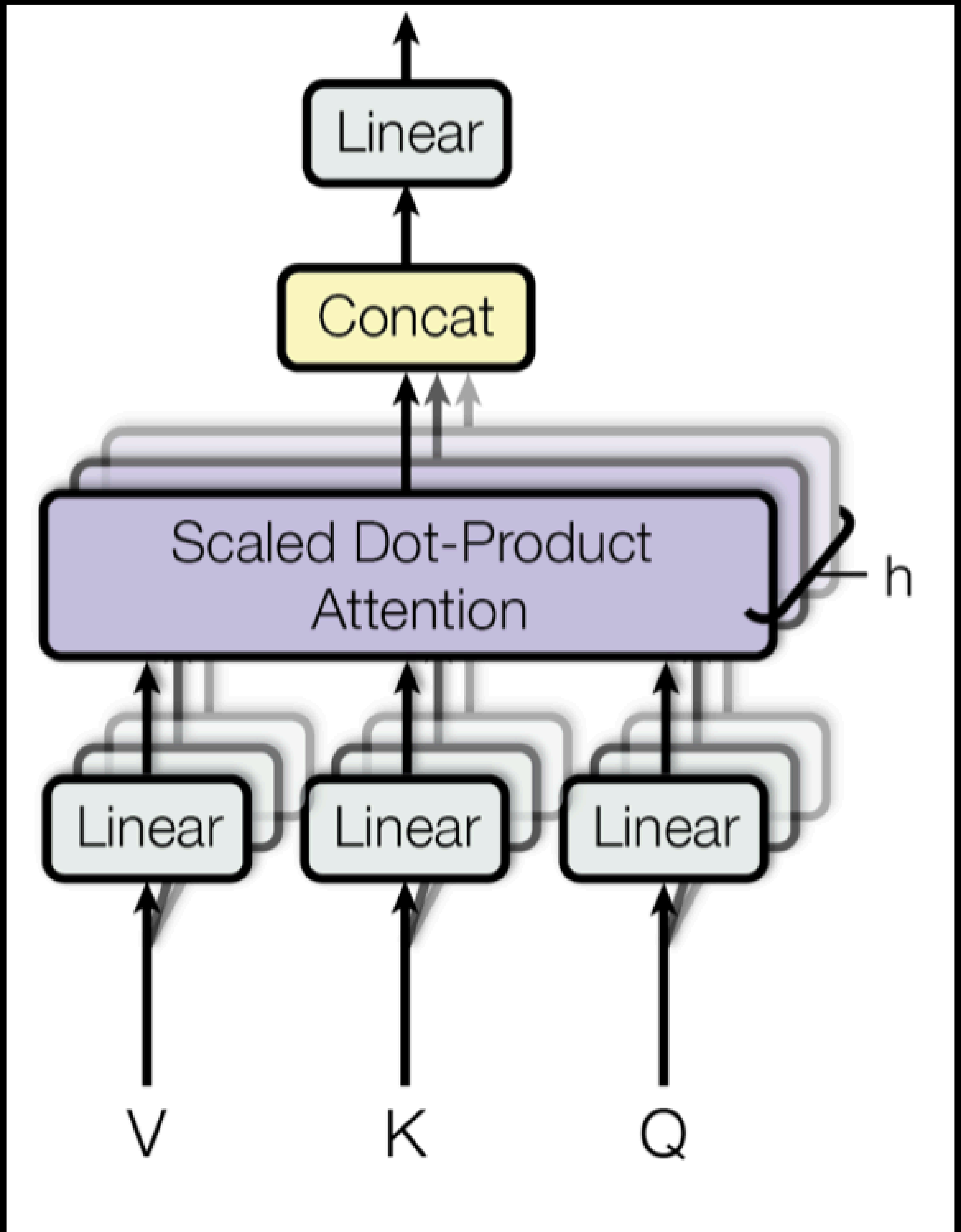
Below, we provide a theoretical analysis of the above spectrum results.

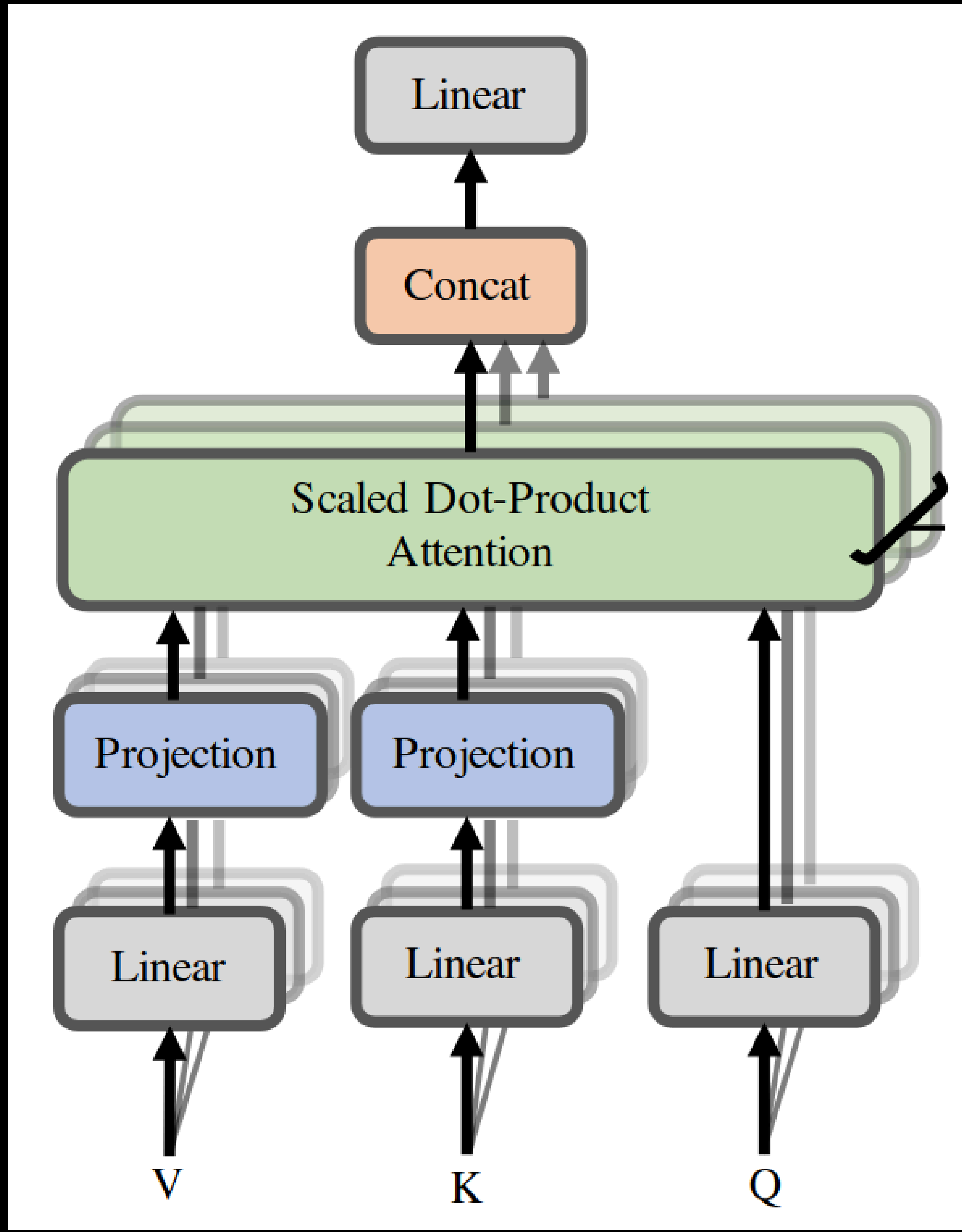
Theorem 1. (self-attention is low rank) *For any* $Q, K, V \in \mathbb{R}^{n\times d}$ *and* $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d\times d}$, *for any column vector* $w \in \mathbb{R}^n$ *of matrix* VW_i^V, *there exists a low-rank matrix* $\tilde{P} \in \mathbb{R}^{n\times n}$ *such that*

$$\Pr(\|\tilde{P}w^T - Pw^T\| < \epsilon\|Pw^T\|) > 1 - o(1) \text{ and } \mathrm{rank}(\tilde{P}) = \Theta(\log(n)), \tag{3}$$

where the context mapping matrix P *is defined in* (2).

Linformer


Linear
Concat
Scaled Dot-Product Attention
h
Linear
Linear
Linear
V
K
Q

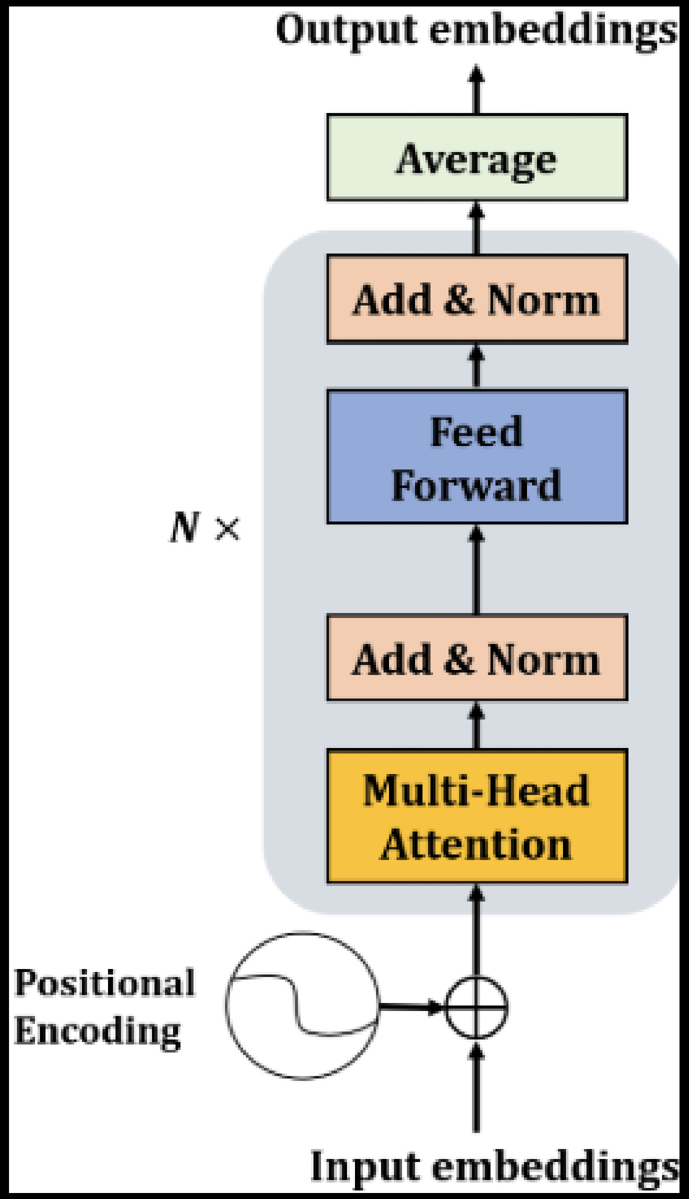

Linear
Concat
Scaled Dot-Product Attention
Projection
Projection
Linear
Linear
Linear
V
K
Q

Transformer: Path Connectivity

Positional Encoding:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}})$$

Hierarchical Processing:



1. Input Processing: Transforms 32x32 pixel images into rich 128-dimensional embeddings.
2. Positional Awareness: Uses sinusoidal encoding to help the model understand spatial relationships between pixels.
3. Multi-Head Attention: 4 attention heads work together to track both local path segments and global connections
4. Hierarchical Processing: 3 transformer layers progressively build understanding from pixel-level to full path recognition.

Perceiver: an Information Bottleneck

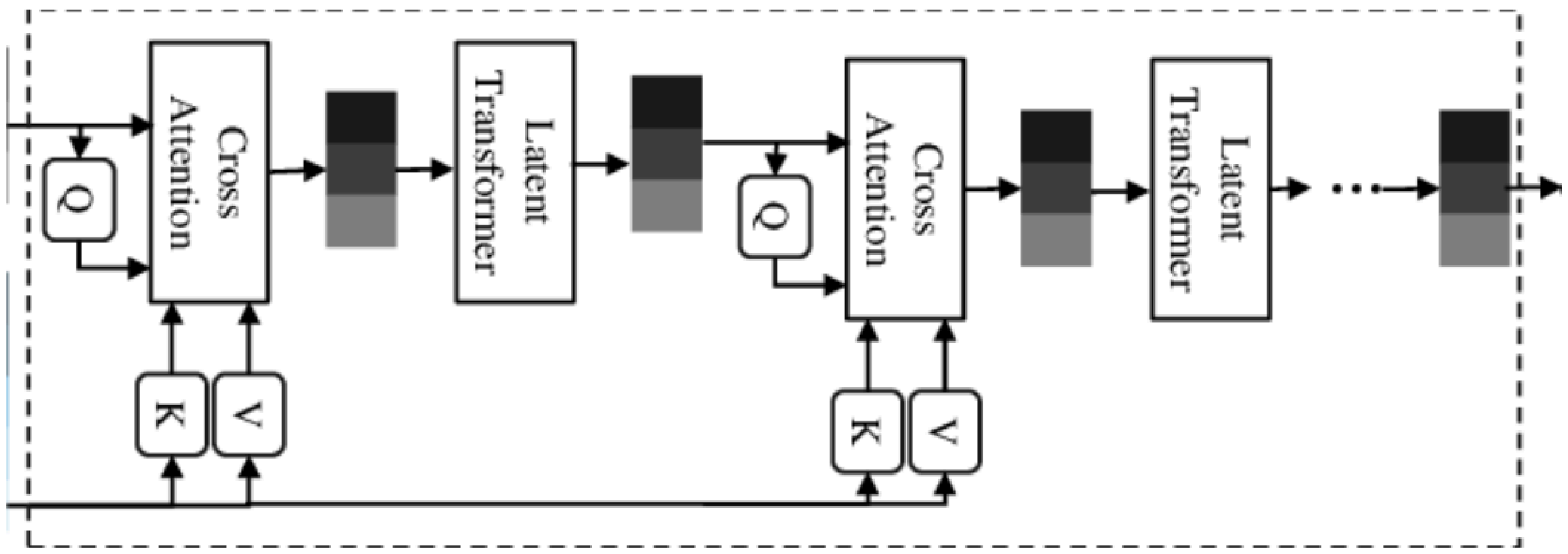


Illustration of the Perceiver architecture.

1. Latent Space Processing Through a Learned Bottleneck.
2. Cross-Attention Mechanism Between Input and Latent Array.
3. Iterative Refinement Through Multiple Processing Steps.

Results

Key Parameters

- ***Loss Function:*** Binary cross-entropy with logits.
- Optimizer: Adam.
- Learning Rate: 0.001.
- Number of Epochs: 100
- Fine Tuning Approach
- Early Stopping Patience: 5
- ReduceLROnPlateau

- ***LSTM and GRU:***
  - Number of Layers: 1.
  - Hidden Size: 128.
- Convolutional Layers (e.g., TCN, CNN, etc.):
  - Kernel Size: 3.
  - Number of Channels: 64.
  - 9 layers of Temporal Blocks
- ***Linformer:***
  - k: 128
  - heads: 2
  - depth: 4
- Embedding and Vocabulary:
  - Vocabulary Size: 256.
  - Embedding Dimension: 64.

Key Parameters

Common Training Parameters

- Loss Function: Binary cross-entropy
- Optimizer: AdamW with weight_decay=0.01
- Base Learning Rate: 3e-4
- Max Epochs: 20
- •Early Stopping Patience: 10
- Gradient Clipping: 1.0
- Mixed Precision Training: 16-bit

Data Processing

- Input Shape: 32x32 → 1024 sequence
- Batch Size: 128
- Train/Val/Test Split: 70/15/15

Transformer Configuration

- Embedding Dimension: 128
- Number of Heads: 4
- Number of Layers: 3
- Feedforward Dimension: 512
- Dropout: 0.1

Transformer Configuration

- Number of Latents: 128
- Latent Dimension: 256
- Self-Attention Layers: 6
- Cross-Attention Layers: 2
- Number of Heads: 8
- Dropout: 0.1

Fine Tuning ?

Hard

Scratch

Model	Accuracy
GRU	50.19
LSTM	49.96

Gradual Fine Tuning

Model	Accuracy
GRU	74.89
LSTM	49.8

RNN / LSTM / GRU

Easy

Embedding

Model	Accuracy
GRU	88.83
LSTM	50.06
RNN	49.93

No Embedding

Model	Accuracy
GRU	50.05
LSTM	49.73
RNN	49.95

GRU no embedding: Not enough Layers ?

Hard

49.8

GRU:
Attention Turbo !!

Normal:

74.89

Self Structured Attention:

73.52

TCN

Barebone	***Standard Improvements***
49.95	86.83

Best Models' Comparison

Model	Accuracy
GRU	74.38
Self Structured Attention	73.52
Linformer *	70.89
TCN *	86.83
ViT	60

Model	Memory (KB)	Speed / Epoch (s)
RNN	168	10
LSTM	466	20
GRU	366	20
GRU 2 Layers	764	10
GRU Self Structured Attention	503	20
RNN No Embedding	70	7
LSTM No Embedding	271	10
GRU No Embedding	204	10
Linformer	5	180
Barebone TCN	1100	180
TCN	1	180

Challenges

Life is too short to try all combinations / architectures
People lie in benchmarks
Data Metadata is lacking
A runtime error can lay to waste hours of training
Did not have time to try Mega / S5 / Big Bird

Lessons Learned

Embedding is the go to technique when dealing with sequences
Ensure Reproducibility by setting randomization seeds
Always use the same architecture as the paper
Start simple - Get complex later
Never underestimate a model
People lie in the benchmarks
Fine Tuning is the MVP
Pytorch Lightning and Transformers

Sources

Linformer: Self-Attention with Linear Complexity
https://arxiv.org/pdf/2006.04768v3

CLASSIFICATION OF LONG SEQUENTIAL DATA USING CIRCULAR DILATED CONVOLUTIONAL NEURAL NETWORKS
https://arxiv.org/pdf/2006.04768v3

LONG RANGE ARENA: A BENCHMARK FOR EFFICIENT TRANSFORMERS
https://arxiv.org/pdf/2011.04006