Hierarchical Perceiver

Radek Bartyzal
15.3.2022
GLAMI AI

Motivation

Current state:

- NN architectures usually differ based on the input modality = audio, image, video, …
- e.g. CNNs and Visual Transformers rely on image specific locality assumption
  - => convolutions, splitting the image into a grid

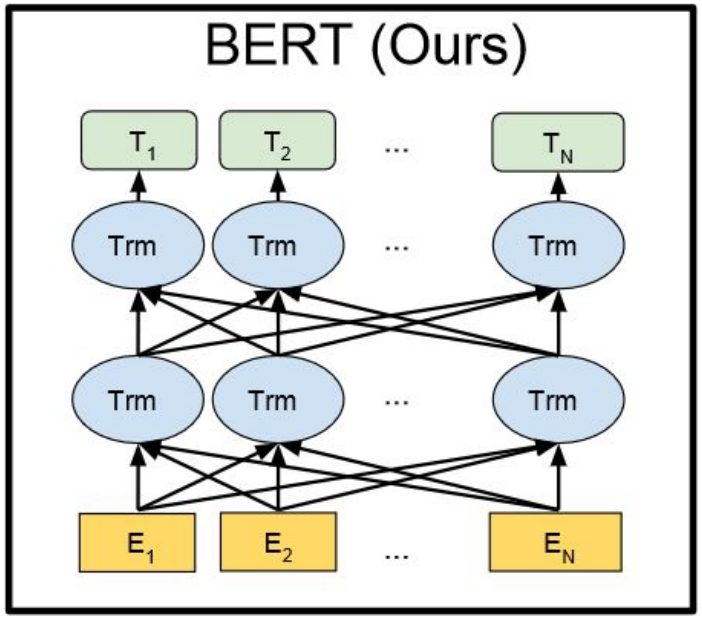
Author’s goal:

- create competitive architecture without relying on these assumptions
  - => attend to the individual pixels in an image
  - => use same architecture for audio, video, 3D point cloud
- enable larger inputs to Transformers

Regular Transformers

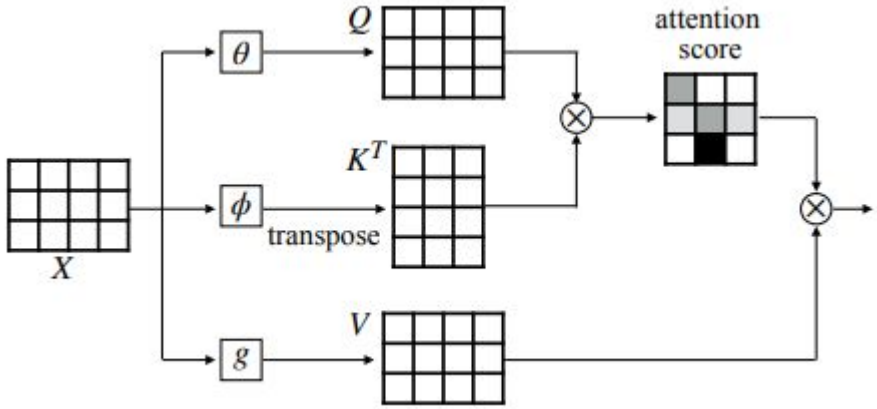
- transform a set of tokens into another set of tokens of same length = 1 transformer layer

- core of the transformer layer = self-attention



Self Attention

- M input tokens = e.g. word embeddings
- from each input token, create Query, Key, Value vectors
- dot product Query * Key vectors => attention matrix of MxM
- dot product Att matrix * Value vectors => new set of token vectors
- => O(M^2) space, O(M^2 * d) time

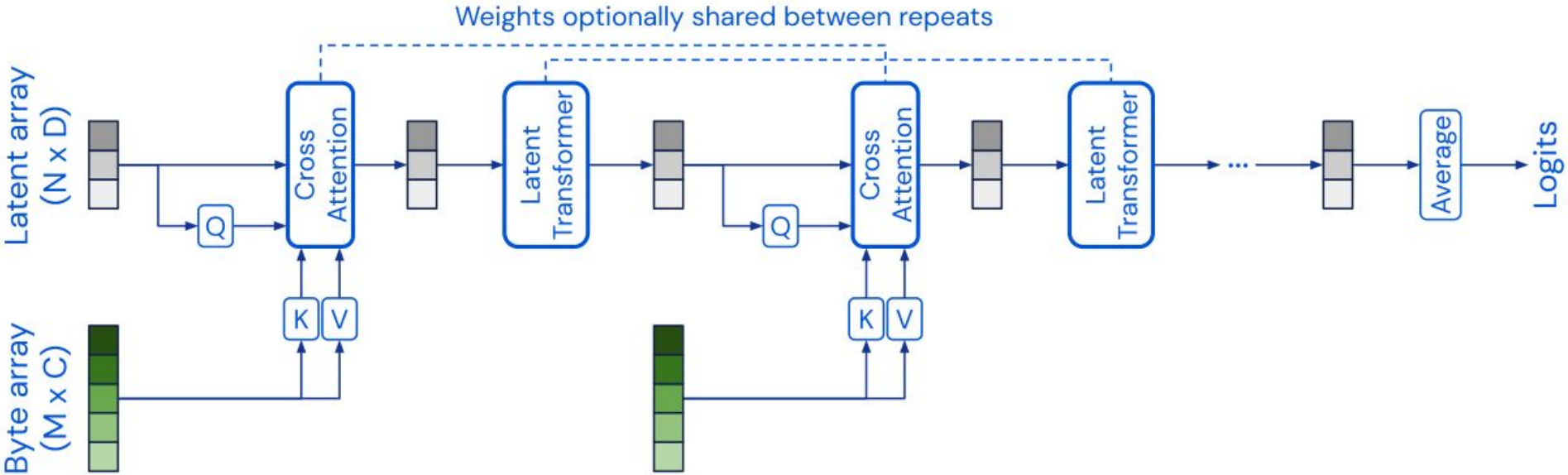


Perceiver architecture

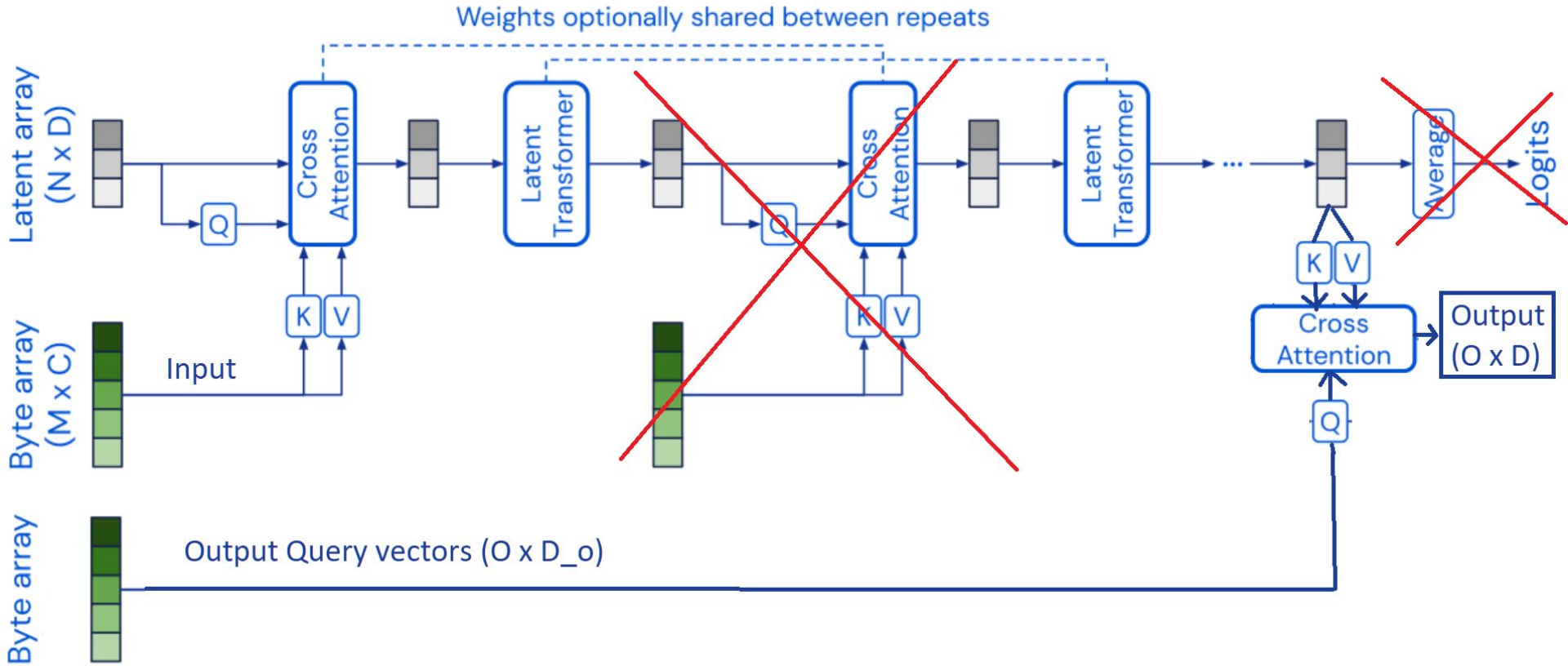
- 224x224 image = 50 000 pixels = M
- => impossible to calculate self-attention with O(50000*50000)
- => use **cross-attention** to inject the input information into the latent representation
- => attention from N latent vectors to M input tokens = O(N*M), N << M

- follow the cross-attention by regular self-attention on the latent space = O(N*N)
- repeat blocks of [cross-attention->regular-latent-attention]

Perceiver architecture



Perceiver IO



Perceiver IO

- autoencoder like structure
- encode = cross-attention of: input -> latent
- decode = cross-attention of: latent -> output
- don’t repeatedly pass in input = does not help that much
- => not a RNN anymore :(

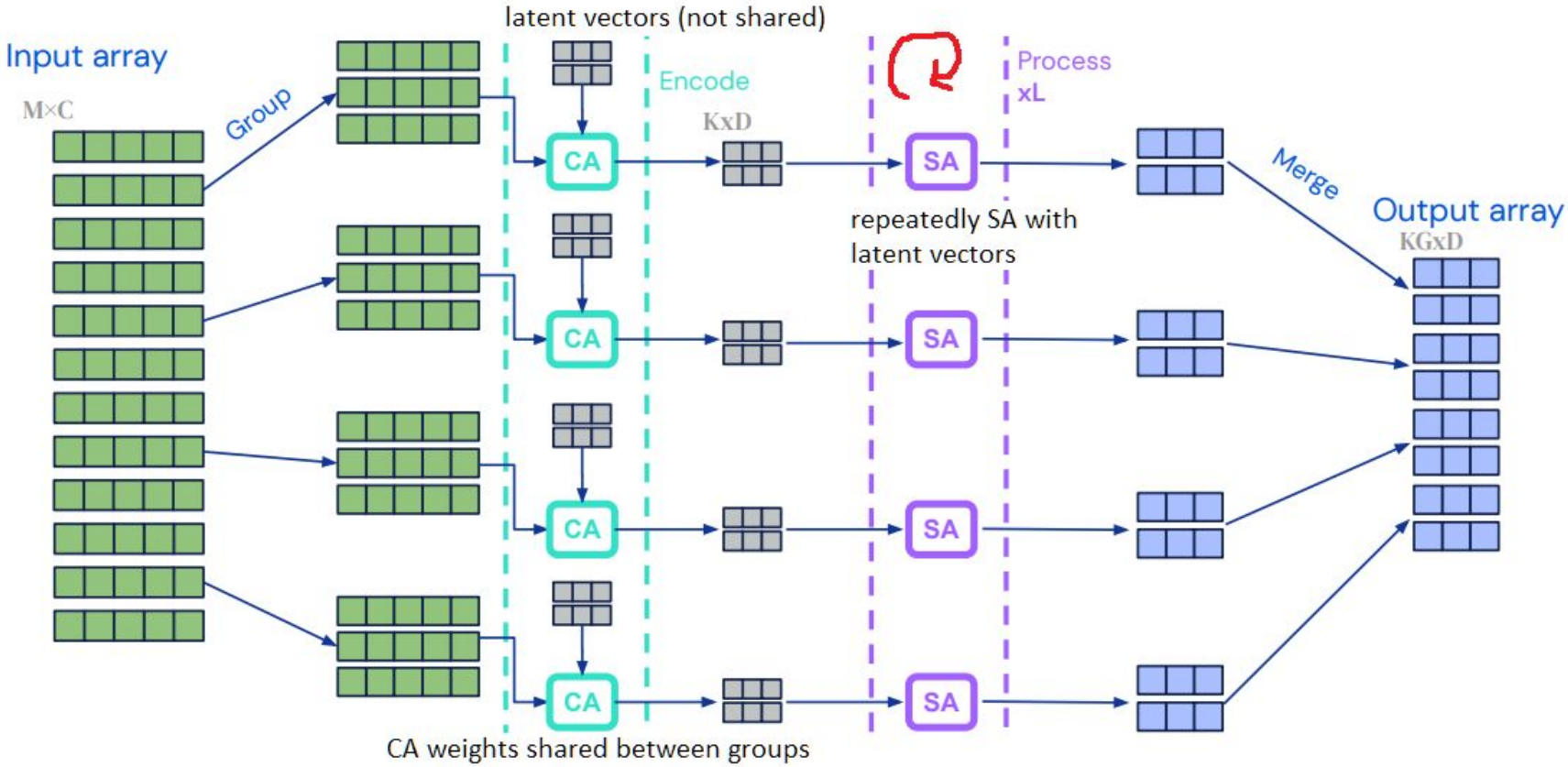
- positional encoding = domain specific = hand crafted
  - 1D = sequence = words
  - 2D = image
  - 3D = 3D point cloud
  - 4D = video

Hierarchical Perceiver Motivation

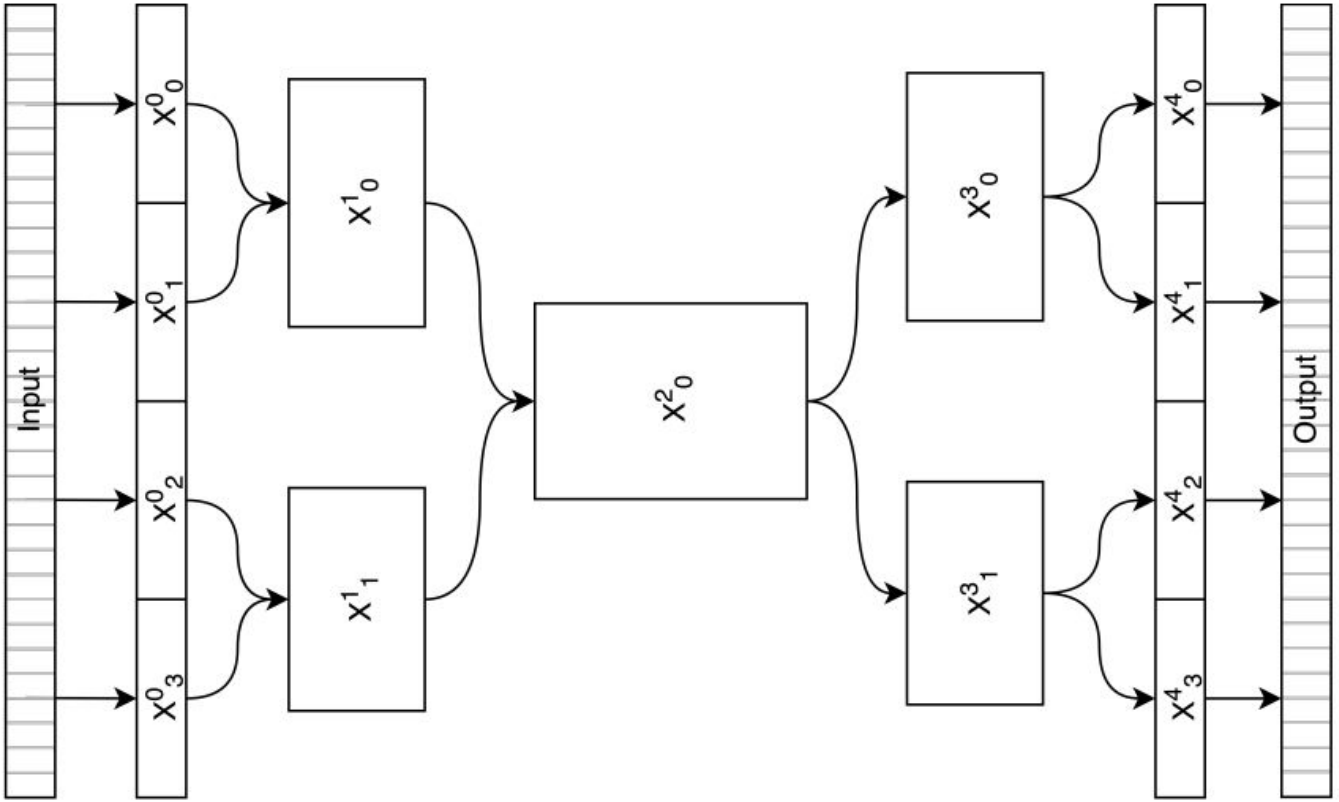
- Perceiver IO still cannot process large inputs = video, hi-res images

- => introduce locality again by grouping the flattened input
  - flatten preserves some locality = use it
- => split input to **G groups**, each with **M/G tokens**
- => process each group separately by Perceiver-like arch
- + weight share between groups except for the latent vectors

Hierarchical Perceiver (HiP) Single Block



HiP Blocks arranged in hierarchy = AutoEn. of AutoEns



Positional Embeddings

- ViT uses learned positional embedding
- works well for small number of inputs = 256 patches
- **negative**: high dimensional

- Perceiver has 1000s of inputs => learned pos. embs. don’t work as well
- => uses Fourier pos. emb.
- their **negatives**:
  - must have correct modality = 1D, 2D, 3D
  - are memory intensive = large dimensionality

- => use Learned pos. embs. with small dimensionality

Learned Positional Embeddings

- training them with classification task as ViT is not effective

- => train them using random, uniform masked auto-encoding
- dropout the masked part of input
- => query the final layer of the hierarchical decoder with the learned positional embeddings corresponding to masked inputs

Experiments

- **HiP 16**: encoder has 16-4-1-1 groups = 4 blocks
- 32 dim positional embeddings

- **HiP-256**: encoder has 256-64-16-4-1-1 groups = 6 blocks
- 16 dim positional embeddings
- more scalable
- more params

Experiments: ImageNet

Image res.	224	384	512	1024	2048
ResNet-50	46.0	20.7	12.5	3.4	OOM
ResNet-101	30.8	14.2	8.7	2.4	OOM
Perceiver IO	2.9	2.4	2.0	OOM	OOM
HiP-16-Fourier	8.6	6.2	4.3	OOM	OOM
HiP-256-Fourier	6.7	4.6	4.2	OOM	OOM
HiP-16	10.8	8.9	7.0	2.9	OOM
HiP-256	8.9	8.1	7.7	4.7	0.9

ImageNet

- HiP 256:
  - worse acc
  - smaller pos. embs
  - better scalability
- Learned pos. features
  - Masked AE approach clearly better

Model	Acc.	Params
ConvNet baselines		
ResNet-50 (He et al., 2016)	78.6	26M
NFNet-F6+SAM (Brock et al., 2021)	86.5	438.4M
Transformer baselines		
ViT-B/16 (Dosovitskiy et al., 2021)	77.9	86M
Swin-B (Liu et al., 2021)	83.5	88M
w/ 2D Fourier features		
Perceiver	78.6	42.1M
Perceiver IO	79.0	48.4M
w/ learned position features		
Perceiver (learned pos)	67.6	55.9M
Perceiver IO (learned pos)	72.7	62.3M
HiP-16	81.0	97.9M
HiP-256	79.9	102.3M

Pos. Emb. Ablations

- Masked AE approach:
  - better results with much smaller dimensionality
- Shuffled pixels:
  - = destroyed locality
  - => 5% hit to acc.

Model	Fourier	MAE	**Acc.**
HiP-16	✓	✓	78.7
HiP-16	✓	✗	78.8
HiP-16	✗	✗	70.1
HiP-256	✓	✗	76.9
HiP-256	✗	✗	68.1
HiP-16 shuffled pixels	✗	✓	76.3
HiP-16 shuffled pixels	✗	✗	68.8
HiP-16	✗	✓	81.0
HiP-256	✗	✓	79.9

AudioSet = audio + video

- Perceiver preprocessed the inputs into audio and video patches (e.g. by concatenating pixels within each 2x8x8 spacetime volume into a single vector)
- HiP: directly feed in individual pixels and raw audio magnitudes

Model	Modalities	#inputs	mAP
Perceiver	A+V	13,024	43.5
HiP-256	A+V	524,288	43.1
Perceiver	A	480	38.3
HiP-256	A	122,880	41.3

Conclusion

- Perceiver and Transformer models:
  - can learn any connectivity pattern
  - because they use global attention
  - BUT dense connectivity = hard to scale
- ConvNets:
  - extremely local operations
  - => very scalable
  - BUT hand-designed structure for different modalities
- HiP:
  - loose, modality-agnostic local connectivity
  - room for soft-connectivity learning in encoder + decoder
  - most of its processing is still fully global and happens at HiP's bottleneck

Conclusion

- positional embeddings cannot be learned for high-resolution signals using global classification losses alone, which is not the case for low resolution signals

- Why? Nobody knows…

Sources

- Perceiver: https://arxiv.org/abs/2103.03206
- Perceiver IO: https://arxiv.org/abs/2107.14795
- HiP: https://arxiv.org/abs/2202.10890