Hiera: A Hierarchical Vision Transformer without the Bells and Whistles

Paper by: Chaitanya Ryali, Yuan-Ting Hu, Daniel Bolya, Chen Wei, Haoqi Fan, Po-Yao Huang, Vaibhav Aggarwal, Arkabandhu Chowdhury, Omid Poursaeed, Judy Hoffman, Jitendra Malik, Yanghao Li, Christoph Feichtenhofer

Facebook AI Research

Transformers

* Deep learning architecture published in 2017 by Google Brain research group

*Encoder-Decoder type architecture that uses the attention mechanism

*Encoder maps input sequence to internal learned representation

* Decoder takes representation and generates a single output, while being fed previous output

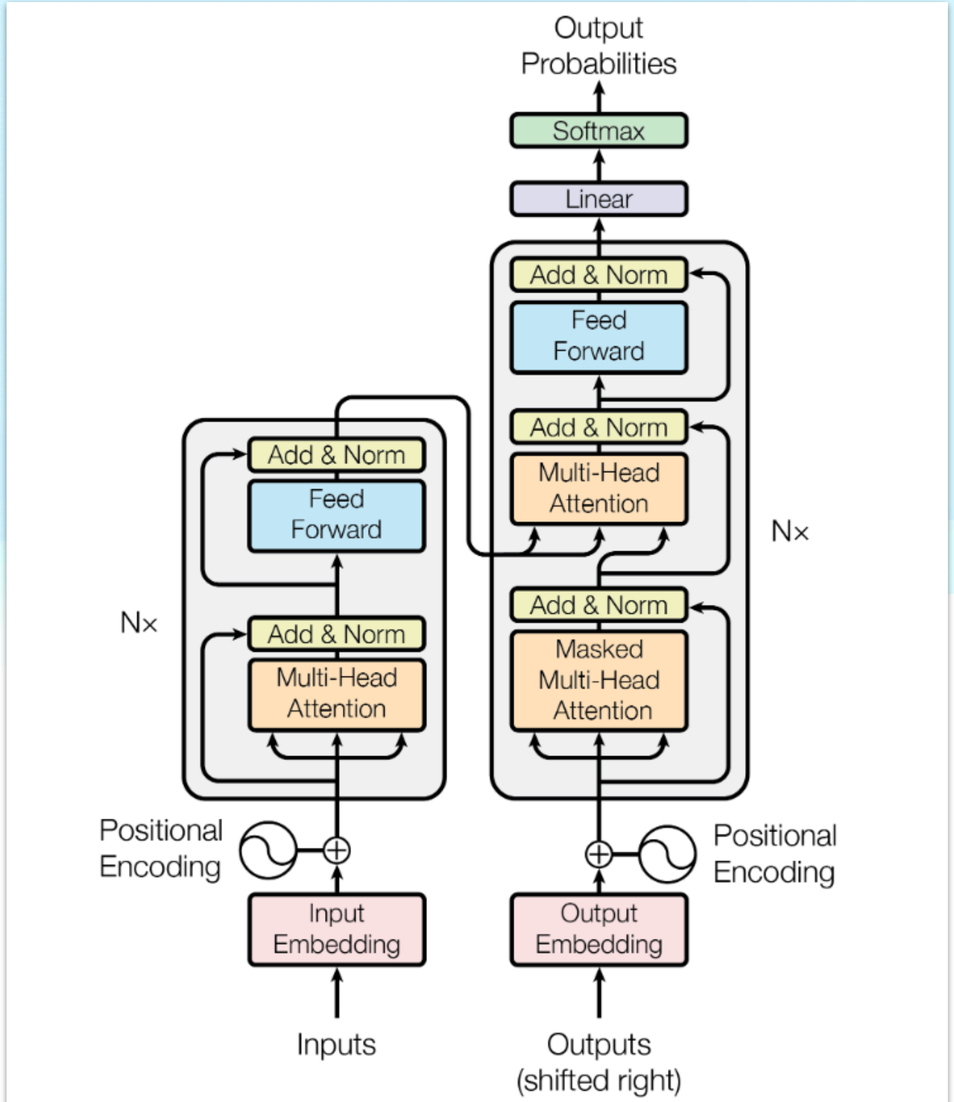


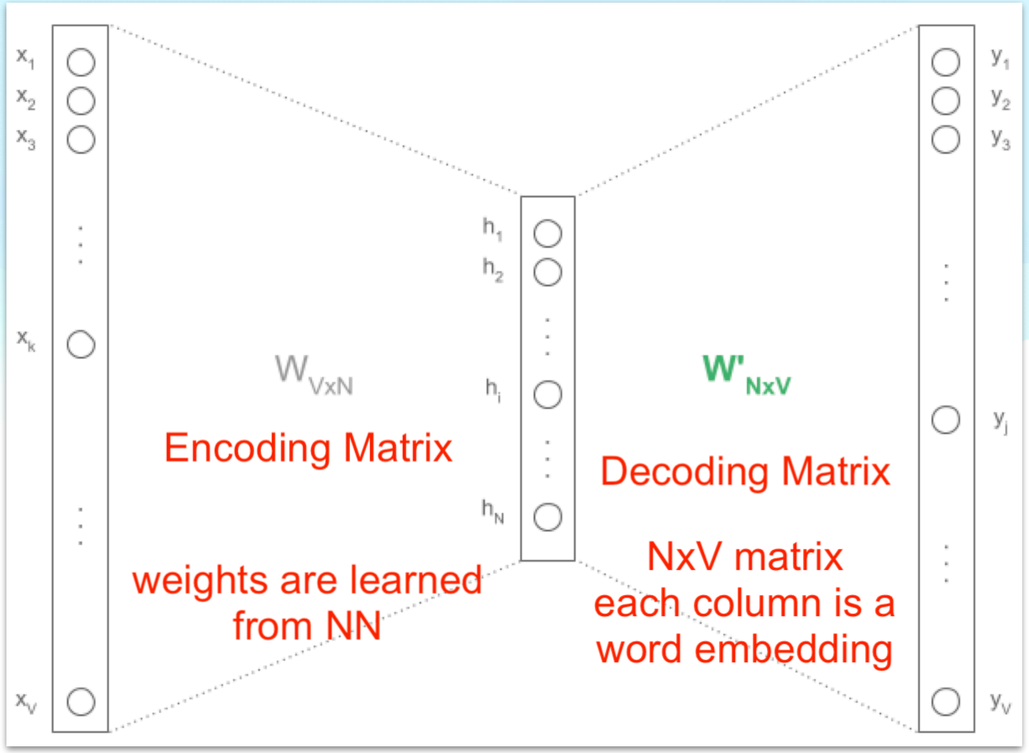
Figure 1: The Transformer - model architecture.

Embedding Input

* Concept used to encode tokens into a numerical format

* Input is a word for example and output is a numerically encoded vector

* Achieved using word/token embedding via a shallow NN, for example word2vec

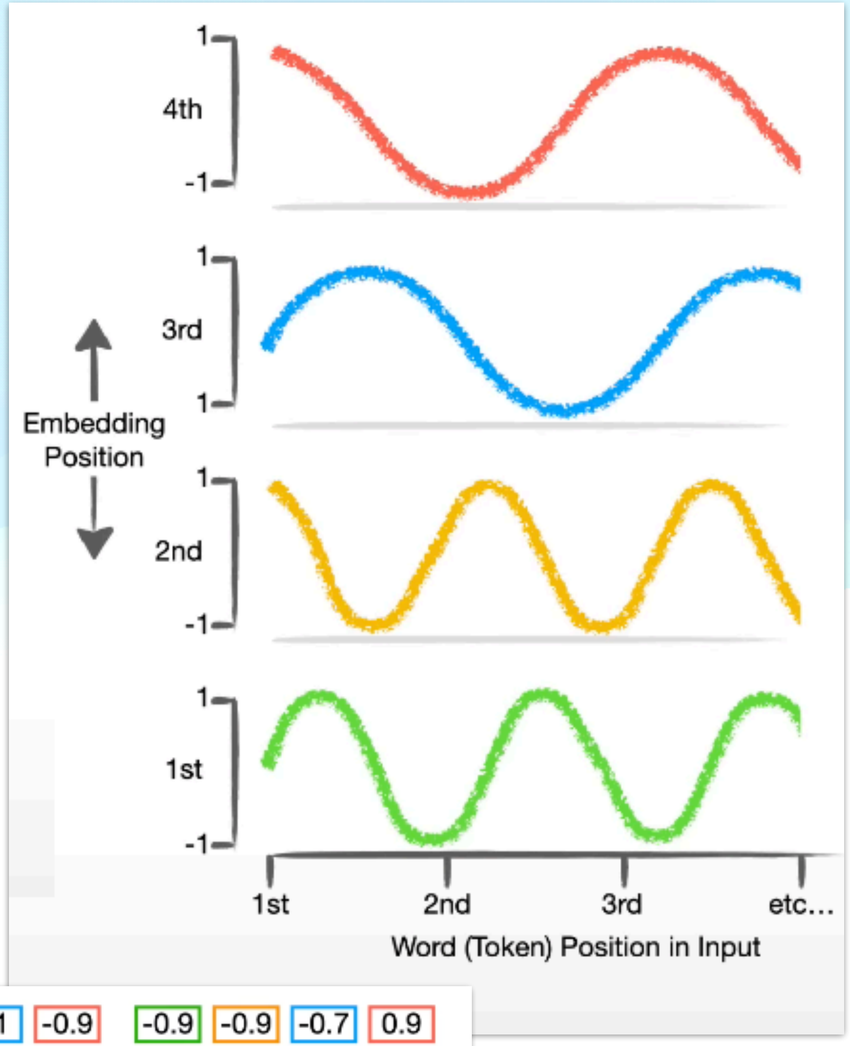


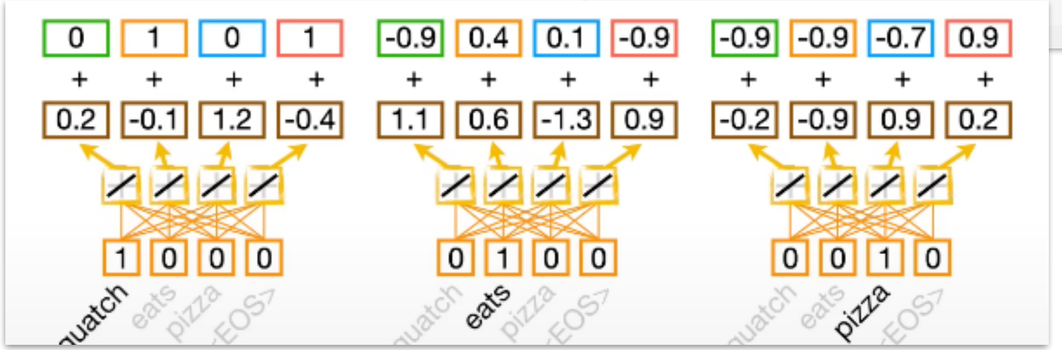
Encoding Position

* Concept used to encode positional significance for tokens

* Order matters for reconstructing images or sentences

* Embedding techniques might use some kind of positional encoding along a sin/cos function





Attention Concept

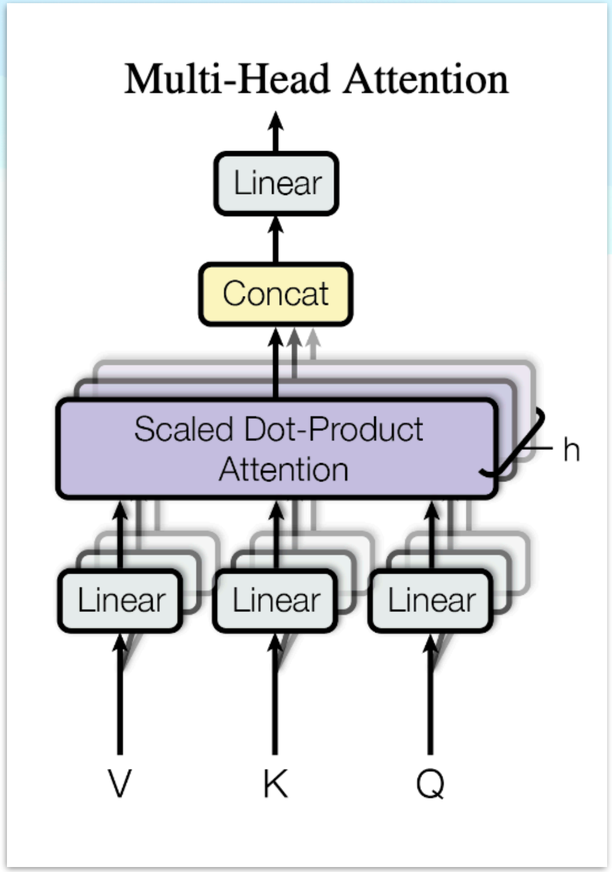
	Hi	how	are	you
Hi	98	27	10	12
how	27	89	31	67
are	10	31	91	54
you	12	67	54	92

* Attention is a measure of a token's relevance to other tokens, including itself

* Key, Value, and Query vectors are learned by a FFNN

* Similarity scores computed from Key, Value, and Query

* Take softmax of these scores to get attention weights



ViT Vision Transformer

* Encoder-Decoder deep network architecture

* Splits image into patches and converts them into numerical vectors (linear embeddings)

* Adds positional embeddings to tokens

* transformer architecture predicts what image might represent

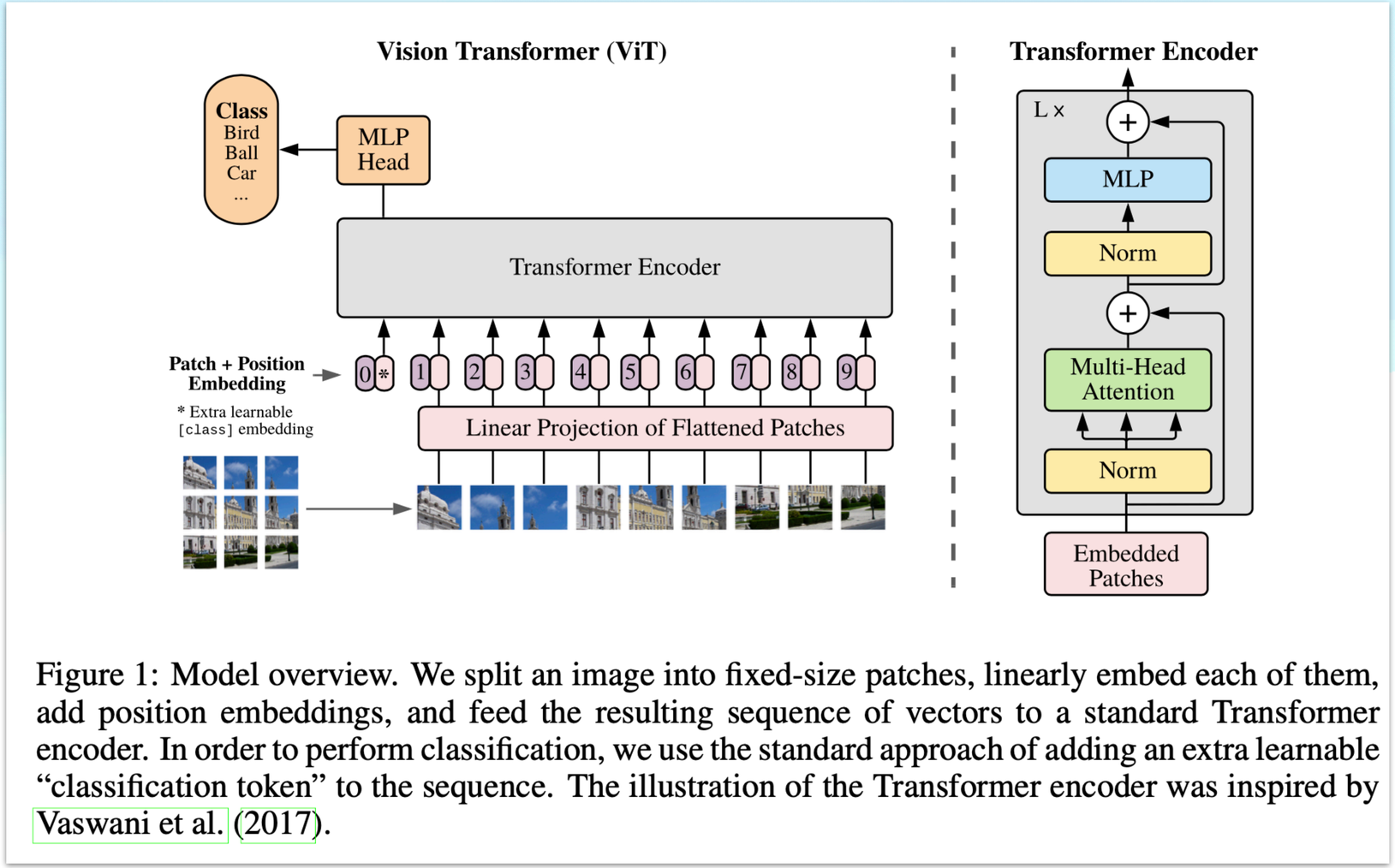


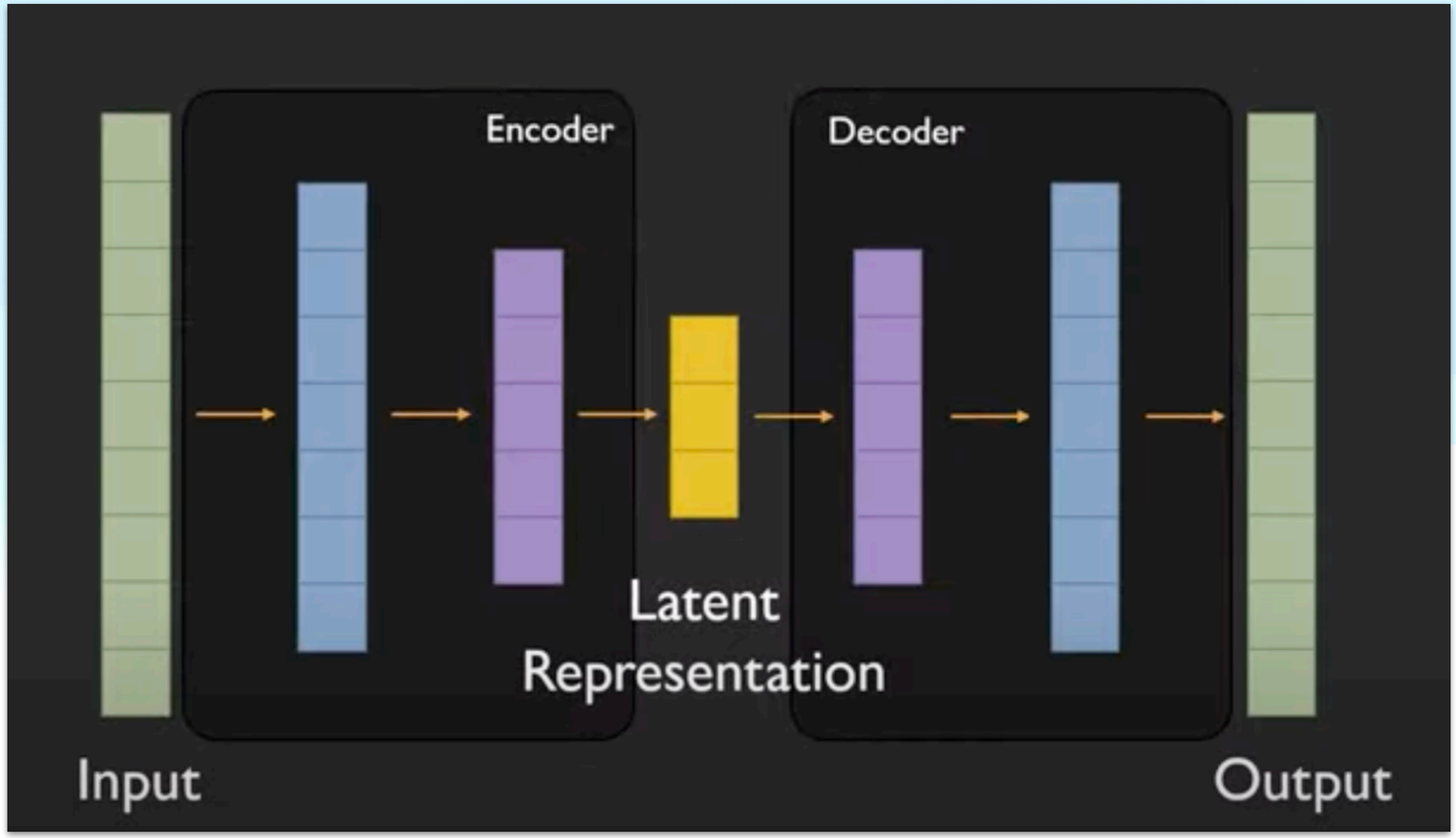
Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Auto Encoders

* (Encoding) Reconstruct the input while being constrained to encode the input into a lower dimensional space

* (Decoding) Reconstruct the input again enforced by an L2 loss between input and output

* Bottleneck in the middle reducing dimension of data is a compressed version of the input data



Masked Auto Encoders

* Take image and divide into non overlapping patches

* Remove a portion of the tokens and encode only the non-masked tokens using ViT encoder

* Positional embeddings remain for non-masked tokens

* Mask tokens used at decoding step and decoder reconstructs missing pixels

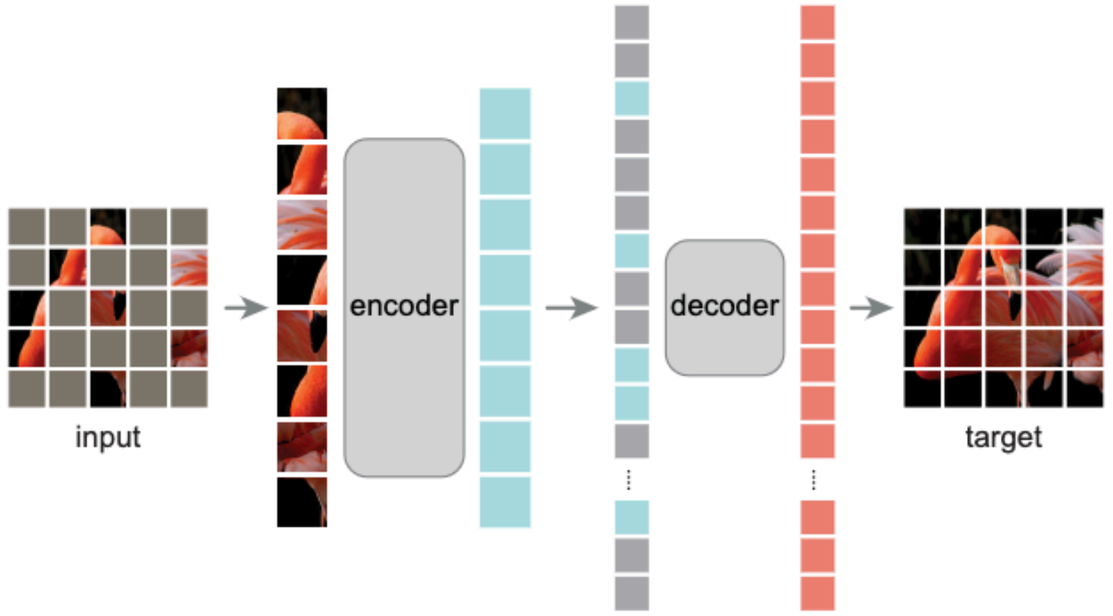


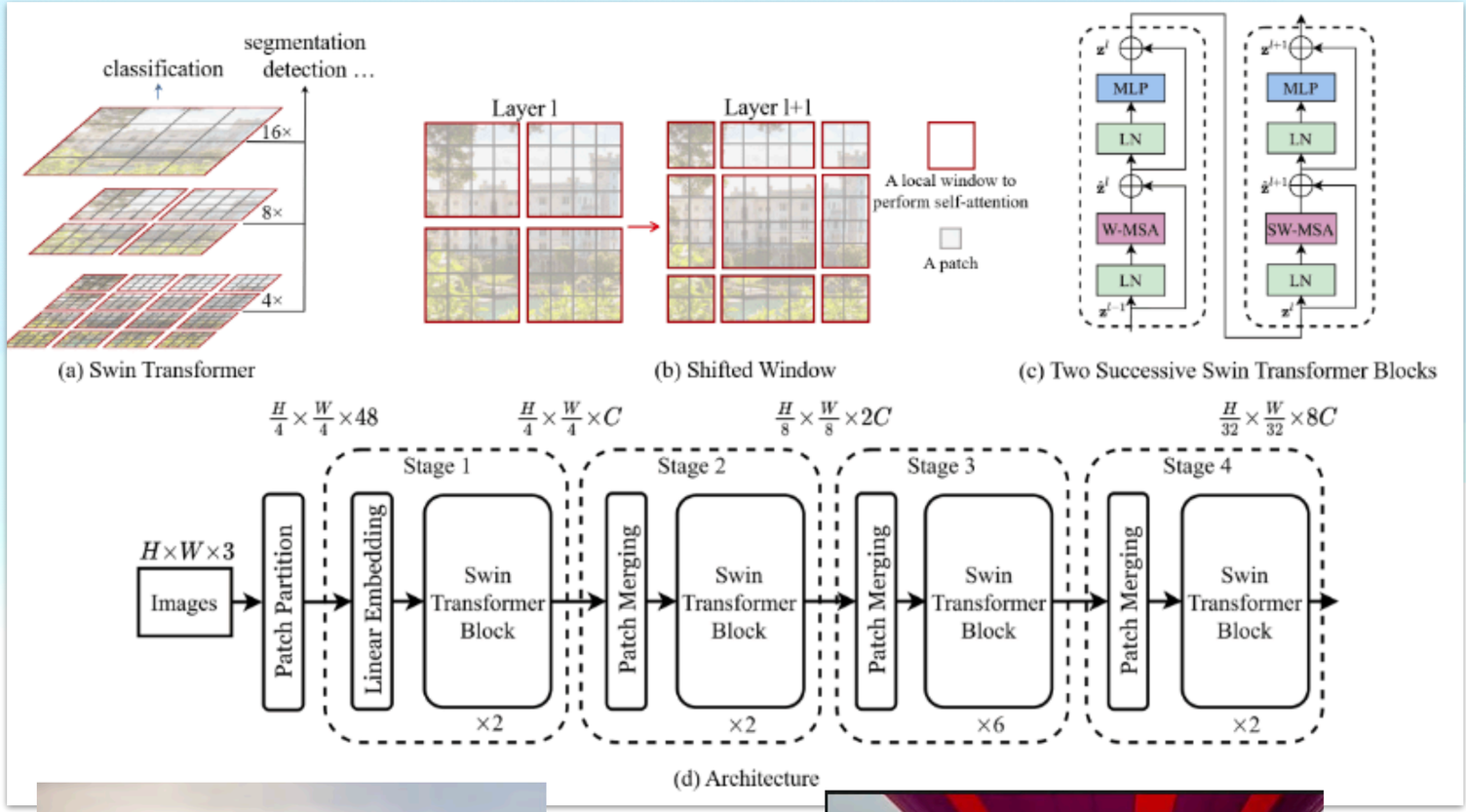
Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

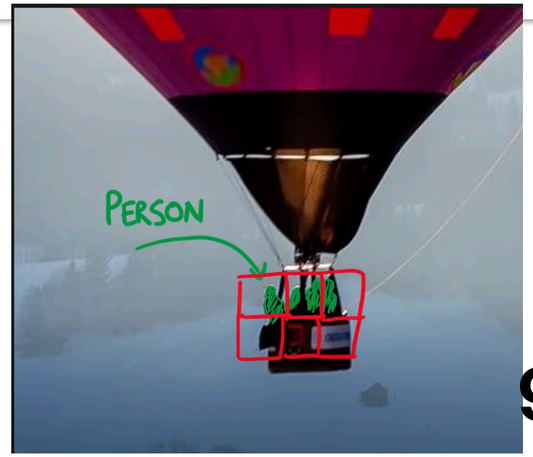
Hierarchical Vision Transformers

* An example is the Swin transformer
  * shifted window transformer

* Why are hierarchical Vision Transformers needed?
  * tokens are sometimes too large for vision tasks

* Hiera Authors outline the complexity adds overhead that slows down these types of models





Hiera Transformer

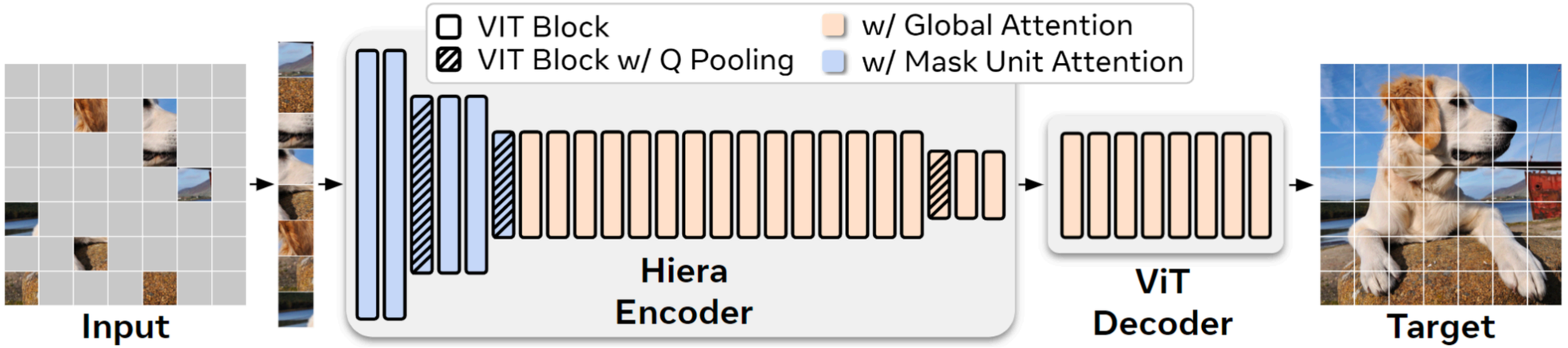


Figure 2. **Hiera Setup.** Modern hierarchical transformers like Swin (Liu et al., 2021) or MViT (Li et al., 2022c) are more parameter efficient than vanilla ViTs (Dosovitskiy et al., 2021), but end up slower due to overhead from adding spatial bias through vision-specific modules like shifted windows or convs. In contrast, we design Hiera to be as simple as possible. To add spatial bias, we opt to *teach* it to the model using a strong pretext task like MAE (pictured here) instead. Hiera consists entirely of standard ViT blocks. For efficiency, we use local attention within "mask units" (Fig. 4, 5) for the first two stages and global attention for the rest. At each stage transition, Q and the skip connection have their features doubled by a linear layer and spatial dimension pooled by a 2×2 maxpool. Hiera-B is shown here (see Tab. 2 for other configs).

*** Uses a simplified version of MViTv2 trained using MAE**

*** Employs work arounds to address common issues with MAE and multi-stage ViTs**

MAE for Hierarchical Models

* MAE pertaining isn't compatible with multi-stage models like Swin

* Some solutions fill in mask token or process un-masked tokens separately which slows down training

* Hiera masks coarser "mask units" and sets kernel size = stride for all convs and prevents them from overlapping

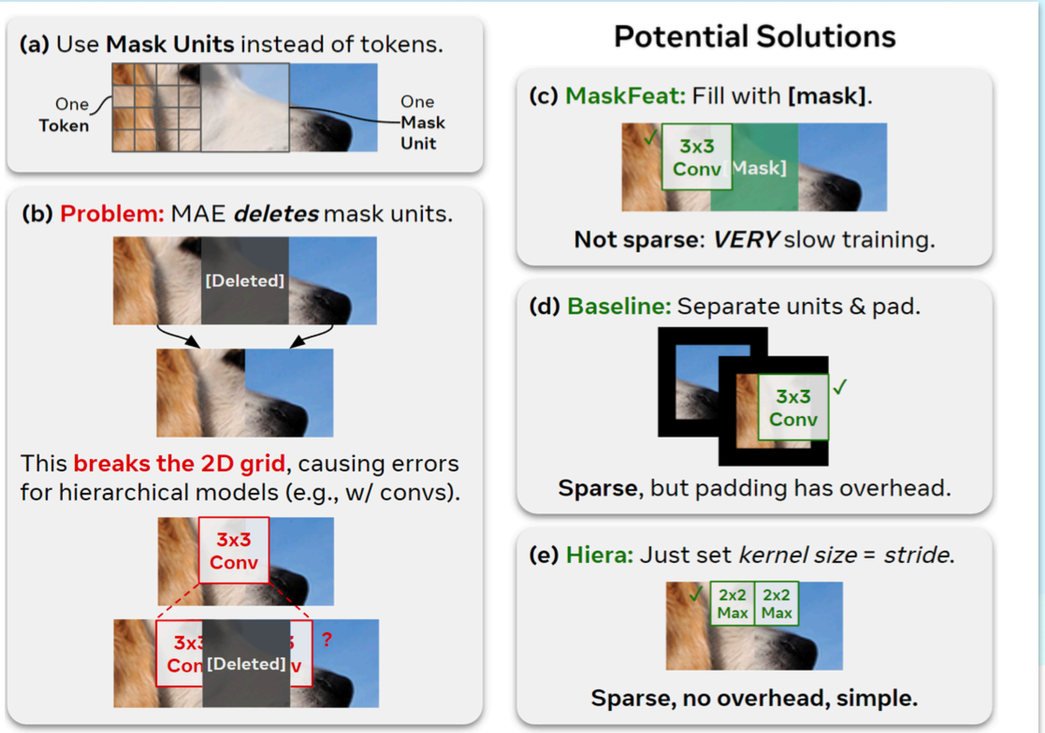


Figure 4. **MAE for Hierarchical Models.** MAE is not compatible with multi-stage models, but we can apply some simple tricks to remedy this. While MAE masks individual tokens, tokens in multi-stage transformers start very small (e.g., 4×4 pixels), doubling size in each stage. (a) Thus, we mask coarser "mask units" (32×32 pixels) instead of tokens directly. (b) For efficiency, MAE is *sparse*, meaning it *deletes* what it masks (a problem for spatial modules like convs). (c) Keeping masked tokens fixes this, but gives up the potential $4 - 10\times$ training speed-up of MAE. (d) As a baseline, we introduce a trick that treats mask units as a separate entities for convs, solving the issue but requiring undesirable padding. (e) In Hiera, we side-step the problem entirely by changing the architecture so the kernels can't overlap between mask units.

Hiera Mask Unit Attention

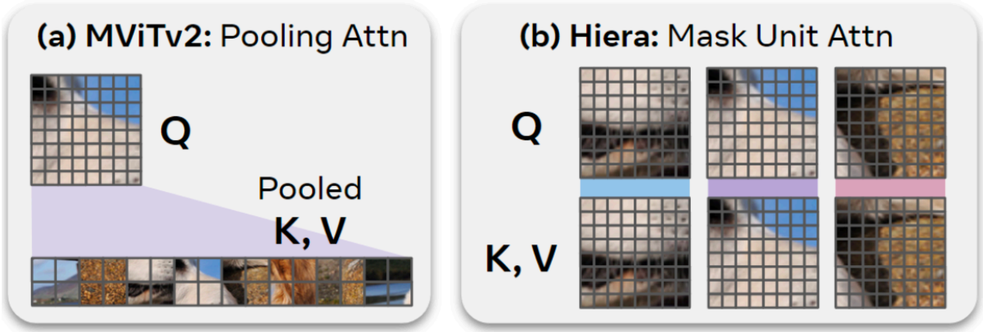


Figure 5. **Mask Unit Attention.** MViTv2 uses pooling attention (a) which performs global attention with a pooled version of K and V. This can get expensive for large inputs (e.g., for video), so we opt to replace this with "Mask Unit Attention" (b) which performs local attention within mask units (Fig. 4a). This has no overhead because we already group tokens into units for masking. We do not have to worry about shifting like in Swin (Liu et al., 2021), because we use global attention in stages 3 and 4 (Fig. 2).

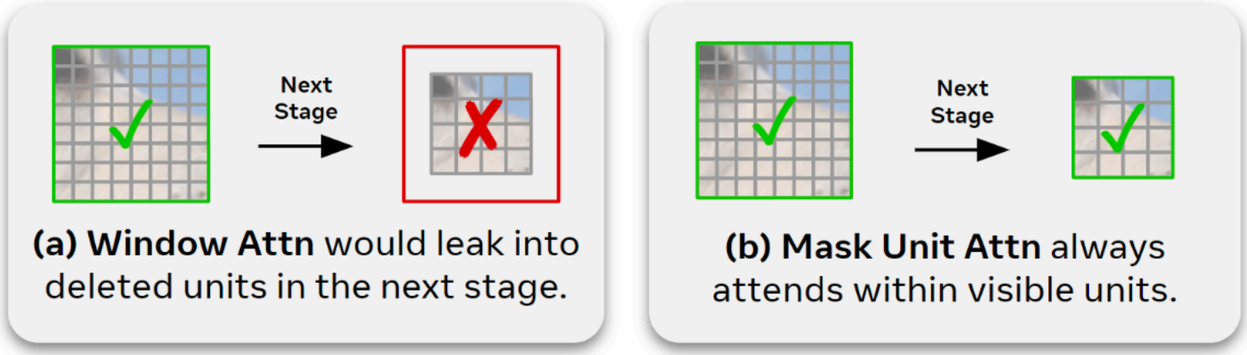


Figure 6. **Mask Unit Attn vs. Window Attn.** Window attention (a) performs local attention within a *fixed* size window. Doing so would potentially overlap with deleted tokens during sparse MAE pretraining. In contrast, Mask Unit attention (b) performs local attention within individual mask units, no matter their size.

*** Perform local attention within mask units instead of global with pooled version of K**

*** Mask Unit attention performs local attention within individual mask units, no matter the number of tokens in the unit**

Experiments

* **Image recognition:** ImageNet-1K is a dataset where ViT has historically struggled, it consists of over 14 million images divided into ~22k different classes (supervised learning)

* **Action recognition:** Kinetics-400 dataset consists of 650k video clips covering 700 classes of human activities (supervised learning)

* **Image and Video reconstruction:** reconstructing masked tokens using transformer decoder blocks (unsupervised learning)

* **Transfer learning:** evaluate transfer learning of K400/K600/K700 pretrained Hiera on action detection using AVA v2.2 and SSv2 video datasets

* **Image Segmentation and Detection**: on COCO dataset

Experiments - Hiera Variants

model	#Channels	#Blocks	#Heads	FLOPs	Param
Hiera-T	[96-192-384-768]	[1-2-7-2]	[1-2-4-8]	5G	28M
Hiera-S	[96-192-384-768]	[1-2-11-2]	[1-2-4-8]	6G	35M
Hiera-B	[96-192-384-768]	[2-3-16-3]	[1-2-4-8]	9G	52M
Hiera-B+	[112-224-448-896]	[2-3-16-3]	[2-4-8-16]	13G	70M
Hiera-L	[144-288-576-1152]	[2-6-36-4]	[2-4-8-16]	40G	214M
Hiera-H	[256-512-1024-2048]	[2-6-36-4]	[4-8-16-32]	125G	673M

Table 2. **Configuration for Hiera variants**. #Channels, #Blocks and #Heads specify the channel width, number of Hierablocks and heads in each block for the four stages, respectively. FLOPs are measured for image classification with 224×224 input. The stage resolutions are $[56^2, 28^2, 14^2, 7^2]$. We introduce B+ for more direct comparison against prior work with slower B models.

* Constructed a collection of variants to compare with similar models in previous work

* Models each have a different number of hierablocks, channels, and attention heads

Experiments - Training Time

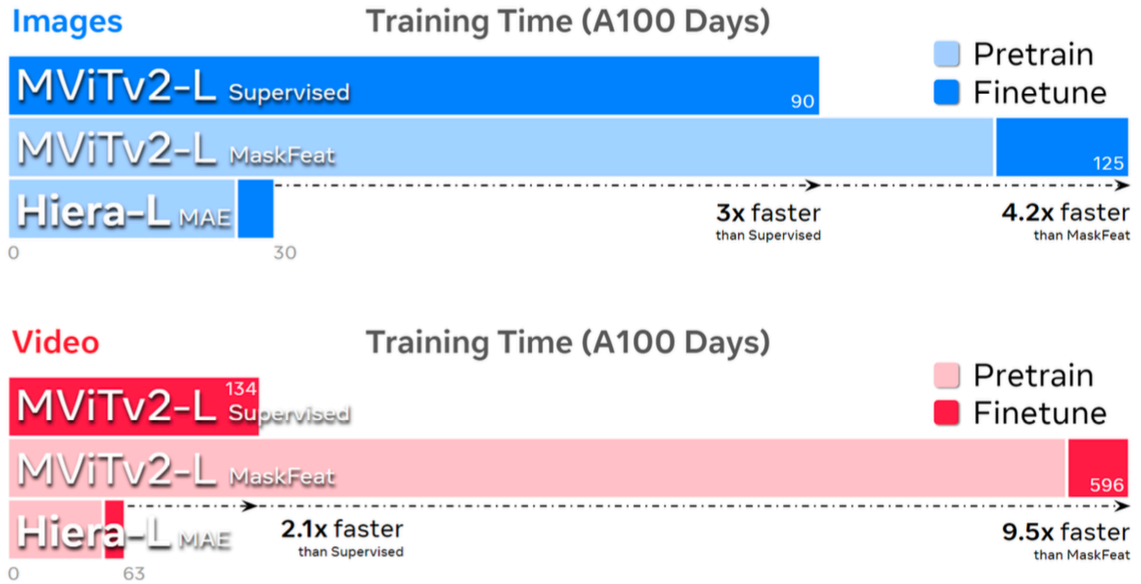


Figure 7. **Training time.** Measured in half precision A100 days. Our Hiera is *significantly* faster to train than MViTv2 due to being more efficient and benefiting from sparse pretraining (as opposed to MaskFeat). Here, supervised uses 300 epochs for ImageNet-1K and 200 for Kinetics-400, while MaskFeat and MAE use 400 for pretraining on images and 800 on video followed by 50 epochs of finetuning for both. Note that Hiera-L at 200 epochs of pretraining (81.8) already outperforms MViTv2-L supervised (80.5) on video, making it 5.6× faster to obtain higher accuracy.

* Hiera was significantly faster than MViTv2 supervised and mask feat variant

* This is due to Hiera being more efficient without the bells and whistles of MViTv2

* Also Hiera benefits from sparse pretraining with MAE

MAE Settings Impact with Heira-L

multi-scale	image	video
✗	85.0	83.8
✔	**85.6**	**85.5**

(a) **Multi-Scale Decoder.** Hiera being *hierarchical*, using multi-scale information for decoding brings significant gains.

mask	image	mask	video
0.5	85.5	0.75	84.9
0.6	**85.6**	0.9	**85.5**
0.7	85.3	0.95	84.4

(b) **Mask ratio.** High masking ratios lead to good performance, with video benefiting from higher masking than image modality.

target	image	video
pixel	85.6	85.5
HOG	**85.7**	**86.1**

(c) **Reconstruction target.** Both pixel and HOG targets result in strong performance.

dpr	image	video
0.0	85.2	84.5
0.1	85.6	85.4
0.2	**85.6**	**85.5**
0.3	85.5	85.2

(d) **Drop path rate.** Surprisingly, we find drop path important during MAE pretraining, especially for video, unlike in He et al. (2022); Feichtenhofer et al. (2022).

depth	image	video
4	85.5	84.8
8	**85.6**	**85.5**
12	85.5	85.4

(e) **Decoder depth.** We find that a deeper decoder than in Feichtenhofer et al. (2022) works better for video.

epochs	image	video
400	85.6	84.0
800	85.8	85.5
1600	**86.1**	86.4
3200	**86.1**	**87.3**

(f) **Pretraining schedule.** Our pretraining follows the same trend as He et al. (2022), benefiting significantly from longer training.

Table 3. **Ablating MAE pretraining with Hiera-L.** For each ablation, we use 400 (800) epochs of sparse MAE pretraining for IN1K (K400) and 50 epochs of dense finetuning unless otherwise noted. Our default† settings are marked in gray. For design choices not ablated here, we find the defaults in (He et al., 2022; Feichtenhofer et al., 2022) to be appropriate. † default pretraining length for the rest of the paper is 1600 (3200) epochs, unless otherwise noted.

Experiments - Simplifying MViTv2

Setting	Image acc.	Image im/s	Video acc.	Video clip/s
MViTv2-L Supervised	85.3	219.8	80.5	20.5
Hiera-L MAE				
a. replace rel pos with absolute *	85.6	253.3	85.3	20.7
b. replace convs with maxpools *	84.4	99.9†	84.1	10.4†
c. delete stride=1 maxpools *	85.4	309.2	84.3	26.2
d. set kernel size equal to stride	**85.7**	369.8	**85.5**	29.4
e. delete q attention residuals	85.6	374.3	**85.5**	29.8
f. replace kv pooling with MU attn	85.6	**531.4**	**85.5**	**40.8**

Table 1. **Simplifying MViTv2.** MViTv2 employs several architectural tweaks to perform well on supervised training. By progressively removing them in Sec. 3.2, we find these bells-and-whistles are *unnecessary* when training with a strong pretext task (MAE). In the process, we create an extremely simple model (Fig. 2) that is accurate while being significantly faster. We report fp16 inference speed for ImageNet-1K and Kinetics-400 on an A100. Our final Hiera-L in gray. *Requires the separate-and-pad trick described in Fig. 4d. †PyTorch's maxpool3d interacts unfavorably with this.

* Progressively removed bells and whistles from MViTv2 model

* Checked accuracy for classification in images and video

* Noticed speed increase and accuracy fluctuations across settings changes

Experiments - Video Data Results

backbone	pretrain	acc.	FLOPs (G)	Param
ViT-B	MAE	81.5	180×3×5	87M
Hiera-B	MAE	84.0	**102**×3×5	**51M**
Hiera-B+	MAE	**85.0**	133×3×5	69M
MViTv2-L	-	80.5	**377**×1×10	218M
MViTv2-L	MaskFeat	84.3	**377**×1×10	218M
ViT-L	MAE	85.2	597×3×5	305M
Hiera-L	MAE	**87.3**	413×3×5	**213M**
ViT-H	MAE	86.6	1192×3×5	**633M**
Hiera-H	MAE	**87.8**	**1159**×3×5	672M

Table 4. **K400 results**. Hiera improves on previous SotA by a large amount, while being lighter and faster. FLOPs are reported as inference FLOPs × spatial crops × temporal clips.

backbone	pretrain	acc.	FLOPs (G)	Param
MViTv2-L	Sup, IN-21K	85.8	377×1×10	218M
MViTv2-L	MaskFeat	86.4	**377**×1×10	218M
Hiera-L	MAE	**88.3**	413×3×5	**213M**
Hiera-H	MAE	**88.8**	1159×3×5	672M

(a) **Kinetics-600** video classification

backbone	pretrain	acc.	FLOPs (G)	Param
MViTv2-L	Sup, IN-21K	76.7	377×1×10	218M
MViTv2-L	MaskFeat	77.5	**377**×1×10	218M
Hiera-L	MAE	**80.3**	413×3×5	**213M**
Hiera-H	MAE	**81.1**	1159×3×5	672M

(b) **Kinetics-700** video classification

Table 5. **K600 and K700 results**. Hiera improves over SotA by a large margin. FLOPs reported as inference FLOPs × spatial crops × temporal clips. Approaches using extra data are de-emphasized.

backbone	pretrain	acc.	FLOPs (G)	Param
K400 pretrain				
ViT-L	supervised	55.7	598×3×1	304M
MViTv2-$L_{40,312}$	MaskFeat	74.4	2828×3×1	218M
ViT-L	MAE	74.0	597×3×2	305M
Hiera-L	MAE	74.7	**413**×3×1	**213M**
Hiera-L	MAE	**75.0**	**413**×3×2	**213M**
SSv2 pretrain				
ViT-L	MAE	74.3	597×3×2	305M
Hiera-L	MAE	74.9	**413**×3×1	**213M**
Hiera-L	MAE	**75.1**	**413**×3×2	**213M**
ViT-L_{32}	MAE	75.4	1436×3×1	305M
Hiera-L_{32}	MAE	**76.5**	**1029**×3×1	**213M**

Table 6. **SSv2 results** pretrained on Kinetics-400 and SSv2. Hiera improves over SotA by a large margin. We report inference FLOPs × spatial crops × temporal clips.

AVA and ImageNet-1K Results

backbone	pretrain	mAP	FLOPs (G)	Param
K400 pretrain				
ViT-L	supervised	22.2	598	304M
MViTv2-$L_{40,312}$	MaskFeat	38.5	2828	218M
ViT-L	MAE	37.0	597	305M
Hiera-L	MAE	**39.8**	**413**	**213M**
ViT-H	MAE	39.5	1192	**633M**
Hiera-H	MAE	**42.5**	**1158**	672M
K600 pretrain				
ViT-L	MAE	38.4	598	304M
MViTv2-$L_{40,312}$	MaskFeat	39.8	2828	218M
Hiera-L	MAE	**40.7**	**413**	**213M**
ViT-H	MAE	40.3	1193	**632M**
Hiera-H	MAE	**42.8**	**1158**	672M
K700 pretrain				
ViT-L	MAE	39.5	598	304M
Hiera-L	MAE	**41.7**	**413**	**213M**
ViT-H	MAE	40.1	1193	**632M**
Hiera-H	MAE	**43.3**	**1158**	672M

Table 7. **AVA v2.2 results** pretrained on Kinetics. Hiera improves over SotA by a large margin. All inference FLOPs reported with a center crop strategy following Fan et al. (2021).

backbone	pretrain	acc.	FLOPs (G)	Param
Swin-T		81.3	**5**	29M
MViTv2-T		82.3	**5**	**24M**
Hiera-T	MAE	**82.8**	**5**	28M
Swin-S		83.0	9	50M
MViTv2-S		83.6	7	**35M**
Hiera-S	MAE	**83.8**	**6**	**35M**
ViT-B		82.3	18	87M
Swin-B		83.3	15	88M
MViTv2-B		84.4	10	**52M**
ViT-B	BEiT, DALLE	83.2	18	87M
ViT-B	MAE	83.6	18	87M
ViT-B	MaskFeat	84.0	18	87M
Swin-B	SimMIM	83.8	15	88M
MCMAE-B	MCMAE	85.0	28	88M
Hiera-B	MAE	84.5	**9**	**52M**
Hiera-B+	MAE	**85.2**	13	70M
ViT-L		82.6	62	304M
MViTv2-L		85.3	42	218M
ViT-L	BEiT, DALLE	85.2	62	304M
ViT-L	MAE	85.9	62	304M
ViT-L	MaskFeat	85.7	62	304M
Swin-L	SimMIM	85.4	**36**	**197M**
MCMAE-L	MCMAE	**86.2**	94	323M
Hiera-L	MAE	86.1	40	214M
ViT-H		83.1	167	**632M**
ViT-H	MAE	**86.9**	167	**632M**
Hiera-H	MAE	**86.9**	**125**	673M

Table 8. **ImageNet-1K** comparison to previous MIM approaches. We de-emphasize approaches using extra data and indicate the source of extra data.

Object detection & segmentation on COCO

backbone	iNat17	iNat18	iNat19	Places365
ViT-B	70.5	75.4	80.5	57.9
Hiera-B	73.3	77.9	83.0	58.9
Hiera-B+	**74.7**	**79.9**	**83.1**	**59.2**
ViT-L	75.7	80.1	83.4	59.4
Hiera-L	**76.8**	**80.9**	**84.3**	**59.6**
ViT-H	79.3	83.0	**85.7**	59.8
Hiera-H	**79.6**	**83.5**	**85.7**	**60.0**
ViT-H_{448}	83.4	86.8	88.3	60.3
Hiera-H_{448}	**83.8**	**87.3**	**88.5**	**60.6**

Table 9. **Transfer learning** on iNaturalists and Places datasets.

backbone	pretrain	AP^{box}	AP^{mask}	FLOPs	params	time
Swin-B	Sup, 21K	51.4	45.4	0.7T	109M	164ms
MViTv2-B	Sup, 21K	53.1	47.4	0.6T	73M	208ms
Swin-B	Sup	50.1	44.5	0.7T	109M	**164ms**
MViTv2-B	Sup	52.4	46.7	**0.6T**	**73M**	208ms
ViTDet-B	MAE	51.6	45.9	0.8T	111M	201ms
Hiera-B	MAE	52.2	46.3	**0.6T**	**73M**	173ms
Hiera-B+	MAE	**53.5**	**47.3**	**0.6T**	92M	192ms
Swin-L	Sup, 21K	52.4	46.2	1.1T	218M	243ms
MViTv2-L	Sup, 21K	53.6	47.5	1.3T	239M	447ms
MViTv2-L	Sup	53.2	47.1	1.3T	239M	447ms
ViTDet-L	MAE	**55.6**	**49.2**	1.9T	331M	396ms
Hiera-L	MAE	55.0	48.6	**1.2T**	**236M**	**340ms**

Table 10. **COCO object detection and segmentation** using Mask-RCNN. All methods are following the training recipe from Li et al. (2022b) and pretrained on ImageNet-1K by default. Methods using ImageNet-21K pretraining are de-emphasized. Test time is measured on a single V100 GPU with full precision.

***Fine tuned mask R-CNN with different pertained backbones on the COCO dataset**

*** Performed segmentation and computed bounding box for object detection and segmentation**

*** Used a training recipe for ViT-Det for image segmentation**

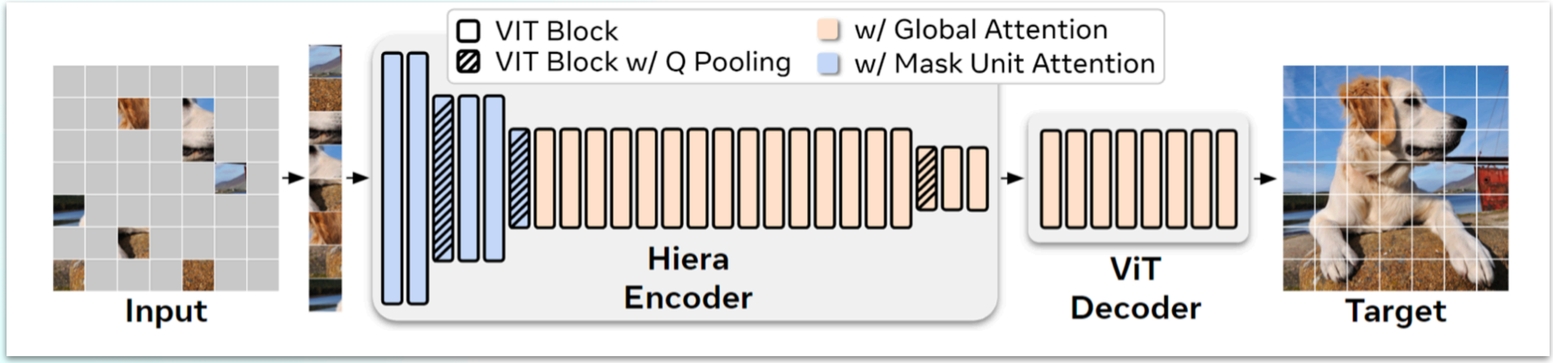
*** Incorporate multi-scale features from Hiera with a Feature Pyramid Network**

Conclusion

* They constructed a simple hierarchical vision transformer by taking an existing one and removing its bells and whistles.

* They supplied the model with spatial bias through MAE pretraining.

* The resulting architecture, Hiera, is more effective than current work on image recognition tasks and surpasses S.O.T.A. on video tasks.



Questions?