MACHINE LEARNING
@ BERKELEY

Vision Transformers (ViTs)

By: ML@B Edu Team

Announcements

- Quiz 5 due tonight
  - It's on sequence models and transformers
- Quiz 6 released today
  - It will cover concepts from today's (Vision Transformers) and Wednesday's (Multimodality) lecture
- Homework 2 due on friday
  - The autograder is up and running
  - Start as soon as possible if you have not already!

Motivation

- Transformers work well for text → what happens if we use them on images?
- Transformers have some nice properties that could be useful for computer vision
    - ex. scalability, global receptive fields

Recall: Transformer Architecture

Start with text string

1. → text tokens
2. → text embedding vectors (via embedding dictionary)
3. → text/position embedding vectors
4. → stacks transformer layers (self-attention + normalization + residual connections + MLP blocks)
5. → CLS token
6. → attach classification head and do prediction, etc.

Commonly trained with a self-supervised objective (ex. next token prediction)

Problem!

Start with text string

1. → text tokens
2. → text embedding vectors (via embedding dictionary)
3. → text/position embedding vectors
4. → stacks transformer layers (self-attention + normalization + residual connections + MLP blocks)
5. → CLS token
6. → attach classification head and do prediction, etc.

Commonly trained with a self-supervised objective (ex. next token prediction)

Naive Solution (imageGPT)

Paper: “Generative Pretraining from Pixels”

- Pixels are kinda discrete — just treat each color value like a separate word in your vocabulary!
  - Each pixel is commonly represented by a 24 bit value (integers in the range [0, 255] for each of the 3 color channels)
  - Vocab size of 2^24 = 16,777,216!
- Who needs that many colors anyway?
  - Use a 9 bit representation (integers in the range [0, 8] for each of the 3 color channels)
  - Vocab size of 512
- Read pixels from raster order (row by row from left to right) to get input sequence

MACHINE LEARNING @ BERKELEY

Naive Solution (imageGPT)

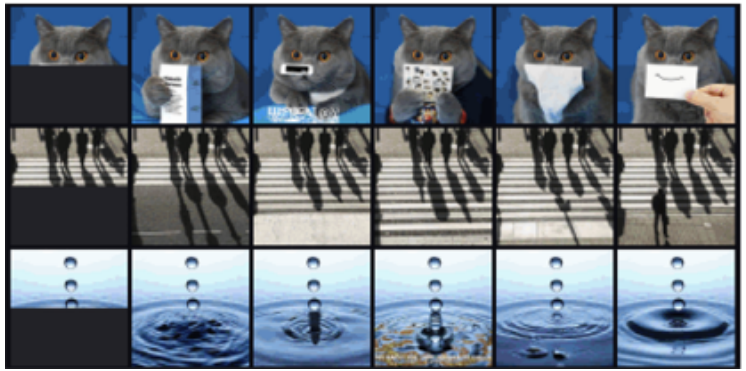
- Another problem: time complexity :(
  - Recall: transformers are O(n^2) w.r.t. input length
  - AND input length is O(n^2) w.r.t. length of each side
  - 256 x 256 image => 65536 pixels
  - For reference, BERT only has a max length of 512 tokens
- Solution: just use smaller images lmao
  - Max size of 64 x 64
- Trained on a similar objective to language models (next pixel prediction instead of next token prediction)

MACHINE LEARNING @ BERKELEY

The good

- Nice image representations
- SOTA on semi-supervised classification
  - Task: classification with limited labeled samples
  - Model: linear classifier on iGPT representations
  - Competitive results with a naive method + **lots of compute**
- Nice image generations
  - Effective at modeling visual information

EVALUATION	MODEL	ACCURACY	PRE-TRAINED ON IMAGENET W/O LABELS	PRE-TRAINED ON IMAGENET W/ LABELS
CIFAR-10 Linear Probe	ResNet-152[50]	94.0		✓
	SimCLR[12]	95.3	✓	
	iGPT-L 32x32	**96.3**	✓	
CIFAR-100 Linear Probe	ResNet-152	78.0		✓
	SimCLR	80.2	✓	
	iGPT-L 32x32	**82.8**	✓	



MACHINE LEARNING
@ BERKELEY

The bad

lots of compute

MACHINE LEARNING @ BERKELEY

The bad

- “We train iGPT-S, iGPT-M, and iGPT-L, transformers containing 76M, 455M, and 1.4B parameters respectively, on ImageNet. We also train iGPT-XL, a **6.8 billion parameter transformer**, on a mix of ImageNet and images from the web.”
- “iGPT-L was trained for roughly **2500 V100-days** while a similarly performing MoCo model can be trained in roughly 70 V100-days”
    - For reference, MoCo is another self-supervised model but it has a ResNet backbone that is capable of handling a 224 x 224 image resolution
- All that for only a 64x64 resolution!

MACHINE LEARNING @ BERKELEY

So… why?

- Mostly a proof of concept
- Paradigm of transformers + *m a s s i v e* self-supervised pre-training but applied to a new domain
  - A general method for learning representations
  - Same method, new modes

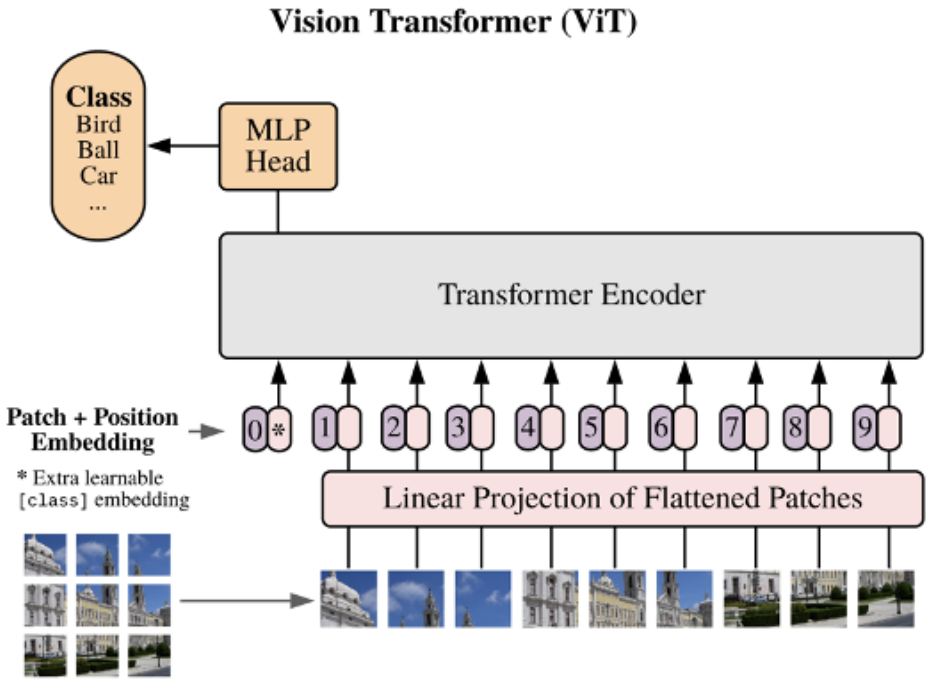
Aside

- There ***are*** ways to make transformers more efficient (architecture-wise)
- BUT recall: a major appeal of using transformers is that they scale well relative to compute
- Transformer architectures are supposed to be simple: self attention is just huge matrix multiplications
    - huge matrix multiplications are good for parallelization
    - want to keep the architecture as simple as possible

A more practical solution

Paper: “An image is worth 16x16 words”

- Rather than quantizing pixels, “downscale” image by splitting it into patches, flattening and then linearly projecting them
  - think: K, Q, V embeddings
  - Each patch becomes a separate sequence token
  - Later architectures used a conv layer instead of patches + linear projections for token embeddings



A more practical solution

- Pretrain at lower resolution, finetune at higher resolution
  - Lower resolution training makes training cheaper / less computationally intensive
  - For more info, check out "Fixing the train-test resolution discrepancy"
- Benchmarked on ImageNet classification
  - Pretrained on massive dataset, then fine-tuned on actual ImageNet
  - The large datasets include ImageNet-22k (a superset of ImageNet with 14M images and 22k classes) and JFT-300M (300M images and 18k classes)

Nice results!

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

MACHINE LEARNING @ BERKELEY

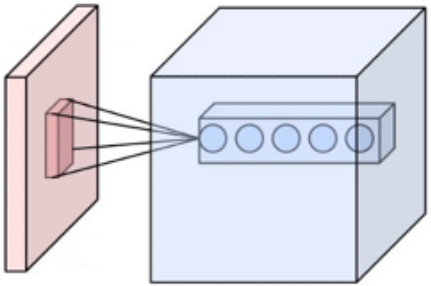
Inductive biases

- Essentially “assumptions” made by the model
  - If you’re using a linear model, you’re assuming the data is linear
- ViTs are more “general” in that they have less inductive biases (more on this later)

Inductive biases

CNNs

- locality
- two-dimensional neighborhood structure
- translation equivariance



Source: https://commons.wikimedia.org/wiki/File:Conv_layer.png

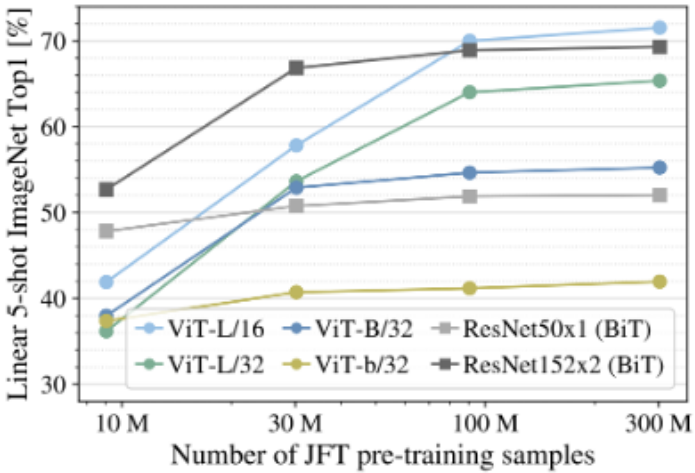
"An image is worth 16x16 words"

- Image patches
- Position embedding interpolation during fine-tuning + upscaling

i.e. MUCH less

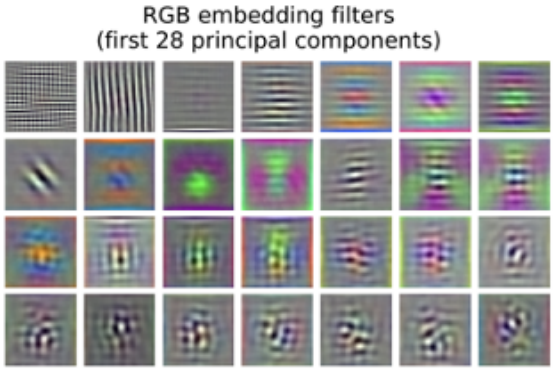
Tradeoff

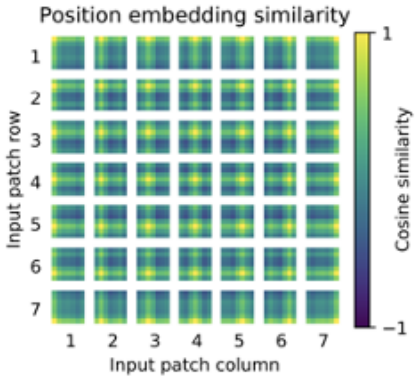
- Performs worse than CNNs when there's not as much data
- Performs better than CNNs when there's lots of data

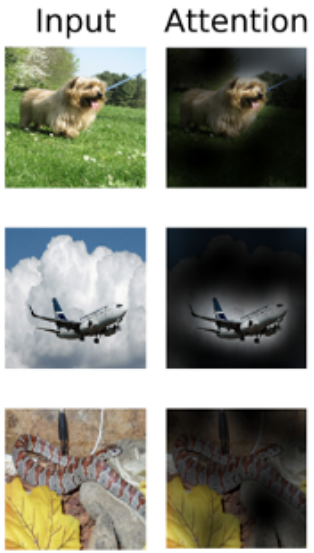


let everything attend to everything

- Why is this tradeoff happening?
- Lack of inductive bias: If we let everything attend to everything during training, the model should learn optimal representations for its task
- i.e. learn biases from data instead of hand engineering them

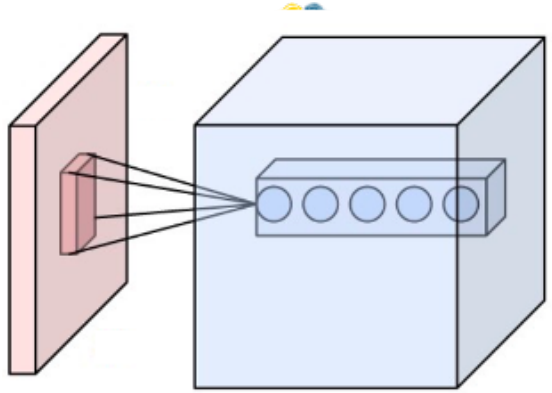


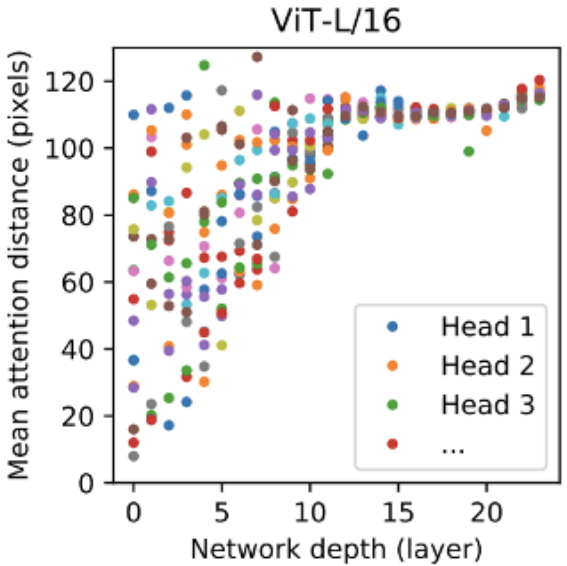




Global receptive field

- CNNs have a receptive field limited by kernel size, especially at earlier layers
- ViTs do not have that restriction
- (Right) Shows the average distance between Q and K for different layers
- Graph shows that ViTs take advantage of larger receptive field





Cons

- Patches are not very fine-grained
- Presents issues for tasks like segmentation, fine-grained classification, etc.

Generality (some takeaways)

- Moving away from inductive biases, towards more general models.
  - Any domain specific stuff should be learned from data. Why?
    - Think:
    - if statements v.s. CNNs → CNNs + supervised learning v.s. transformers + unsupervised learning
      - learned algorithms tend to be better (*if we have enough data)
    - Lets us easily combine domains. If our architecture/learning scheme is general, we can apply it easily to text, images, video, audio etc.
      - Text, images… atari games — all modeled with transformers
      - See: GATO

Generality (some takeaways)

Towards self-supervised and unsupervised learning

- Why?
  - Lesson from “An image is worth 16x16 words” — Transformers need a lot of data to work, but perform very well if you have it
    - They also have the GPU/TPU compatibility that allows them to scale enough to actually handle that amount of data
  - (See: *if we have enough data) We have a lot of ~data~ but not a lot of labeled data
    - Self-supervised and unsupervised objectives let us use unlabeled data!

Transformers give us *Generality*, where biases are learned from data instead of hand crafted.

i.e. let everything attend to everything and if we give it enough data, it'll figure out what it should attend to

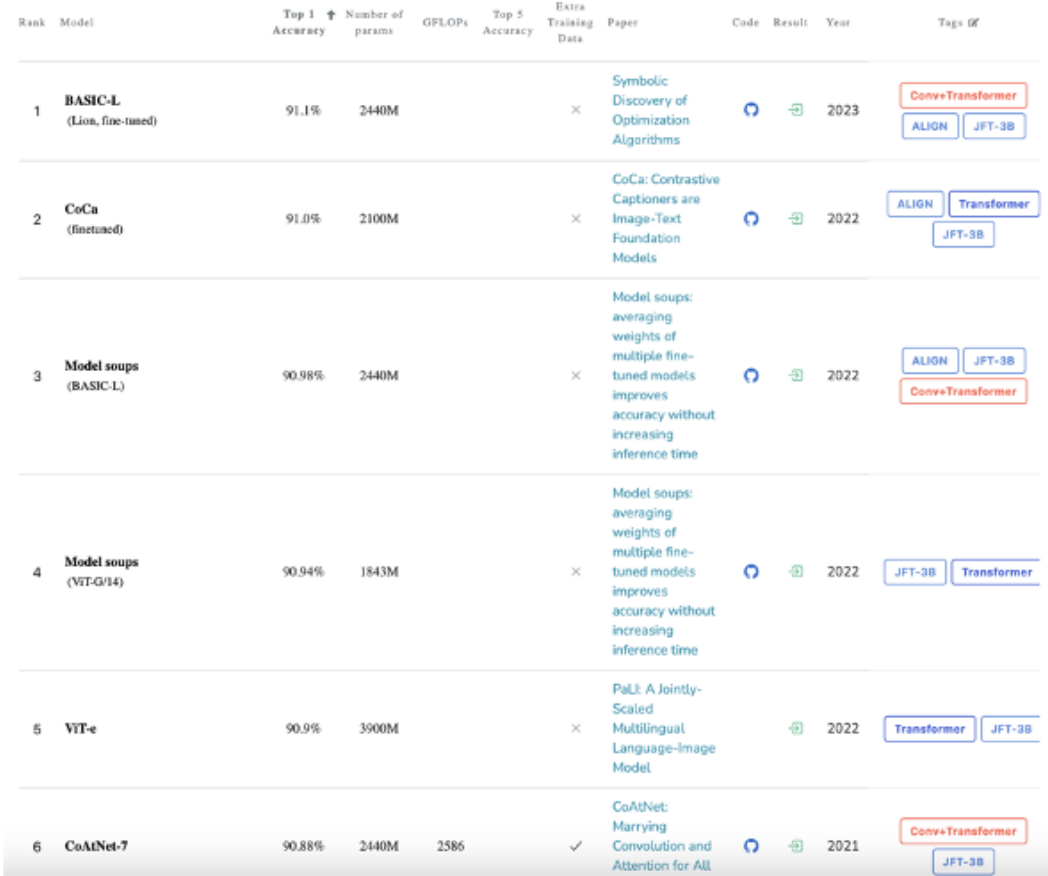
MACHINE LEARNING @ BERKELEY

oh by the way haha — ViTs are also really good!

Notice:

- Parameter count
  - 1 billion +
- “JFT-3B”
  - h u g e pre-training dataset
  - 3 billion images

i.e. transformers allowed people to scale things up!



Rank	Model	Top 1 Accuracy	Number of params	GFLOPs	Top 5 Accuracy	Extra Training Data	Paper	Code	Result	Year	Tags
1	**BASIC-L** (Lion, fine-tuned)	91.1%	2440M			×	Symbolic Discovery of Optimization Algorithms			2023	Conv+Transformer, ALIGN, JFT-3B
2	**CoCa** (finetuned)	91.0%	2100M			×	CoCa: Contrastive Captioners are Image-Text Foundation Models			2022	ALIGN, Transformer, JFT-3B
3	**Model soups** (BASIC-L)	90.98%	2440M			×	Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time			2022	ALIGN, JFT-3B, Conv+Transformer
4	**Model soups** (ViT-G/14)	90.94%	1843M			×	Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time			2022	JFT-3B, Transformer
5	**ViT-e**	90.9%	3900M			×	PaLI: A Jointly-Scaled Multilingual Language-Image Model			2022	Transformer, JFT-3B
6	**CoAtNet-7**	90.88%	2440M	2586		✓	CoAtNet: Marrying Convolution and Attention for All			2021	Conv+Transformer, JFT-3B

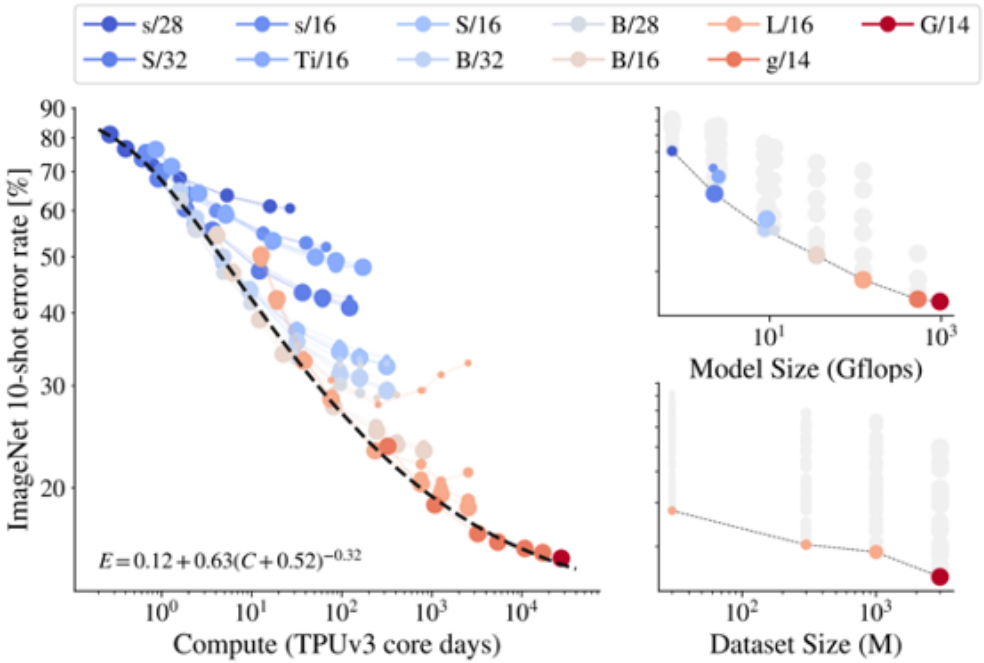
From “cool in theory” to SOTA

MACHINE LEARNING @ BERKELEY

Compute goes brrr

Paper: “Scaling Vision Transformers”

- Same arch as “An image is worth 16x16 words”
- Insights into scaling ViTs
  - Better representation scales with compute time, model size, and dataset size
  - Large models are more sample efficient (i.e. they need less training samples to get the same performance)
- Some engineering stuff to improve training performance/fine-tuning accuracy
- 2B param model got ImageNet SOTA (at the time)



Blue-r = smaller models

Smaller = smaller datasets

Top right = model size bottleneck

Bottom right = dataset size bottleneck

Notice: (Left) error trails off faster with smaller models

Takeaway: if you scale up data, model size, and compute at the same time then you perform better :)

Scaling behavior for vision transformers

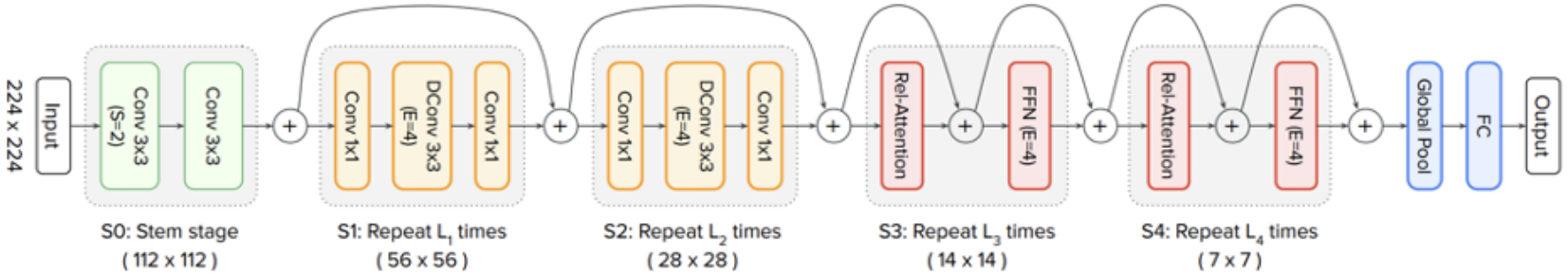
ViTs x CNNs

Paper: “CoAtNet: Marrying Convolution and Attention for All Data Sizes”

- Recall: lack of inductive biases bad when there’s not much data
- Add *some* inductive biases into ViTs
- Two issues with transformers
  - Missing translational invariance
    - -> add a weight that’s only dependent on relative position
  - Quadratic time wrt spatial size
    - CNN style pooling
- Architecture is MBConv blocks + relative attention blocks
  - MBConv: Block of convolutional operations (from MobileNet V2)
- ImageNet SOTA (at the time)

$$y_i^{\text{pre}} = \sum_{j \in \mathcal{G}} \frac{\exp\left(x_i^\top x_j + w_{i-j}\right)}{\sum_{k \in \mathcal{G}} \exp\left(x_i^\top x_k + w_{i-k}\right)} x_j.$$

ViTs x CNNs



Notice:

- Transformers applied later, where the input is smaller
- Input dimension to each block is downsampled (pools before input)

Conclusion

- ViTs (and their associated self-supervised training schemes) as the “next step” in more and more **general** models
  - Less inductive biases and less labels
  - Hand engineered features → CNNs → ViTs
  - Requires a lot of data
- ViTs are super scalable (letting us take advantage of huge datasets)
  - And they perform really well!

MACHINE LEARNING
@ BERKELEY

Lecture Attendance

https://tinyurl.com/fa23-dl4vd

MACHINE LEARNING
@ BERKELEY

Contributors

- Alexander Wan