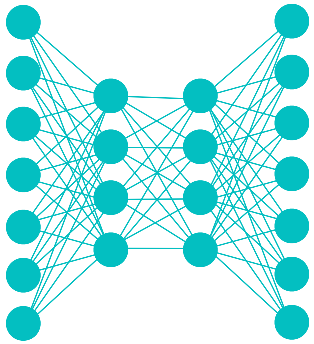
Lecture Notes for

Neural Networks and Machine Learning

Auto-regressive and Vision Transformers



Logistics and Agenda

- Logistics
  - Grading update
  - ICLR Best papers
- Agenda
  - Finish BERT example
  - Decoder Transformers
  - Vision Transformers
  - Paper Presentation
  - Town Hall

Fine Tuning BERT “finish”

20 News Groups

eclarson Eric Larson

```
Main Repository:
     02 BERT Transfer.ipynb
Since we are using hugging face for this, its
better to use PyTorch …

Encoded inputs:
{‘input_ids’: tokens to embed,
 ‘token_type_ids’: segment embed,
 ‘attention_mask’: for causal or non-causal}
```

Auto-regressive Transformers

Ethics and Information Technology (2024) 26:38
https://doi.org/10.1007/s10676-024-09775-5

ORIGINAL PAPER

ChatGPT is bullshit

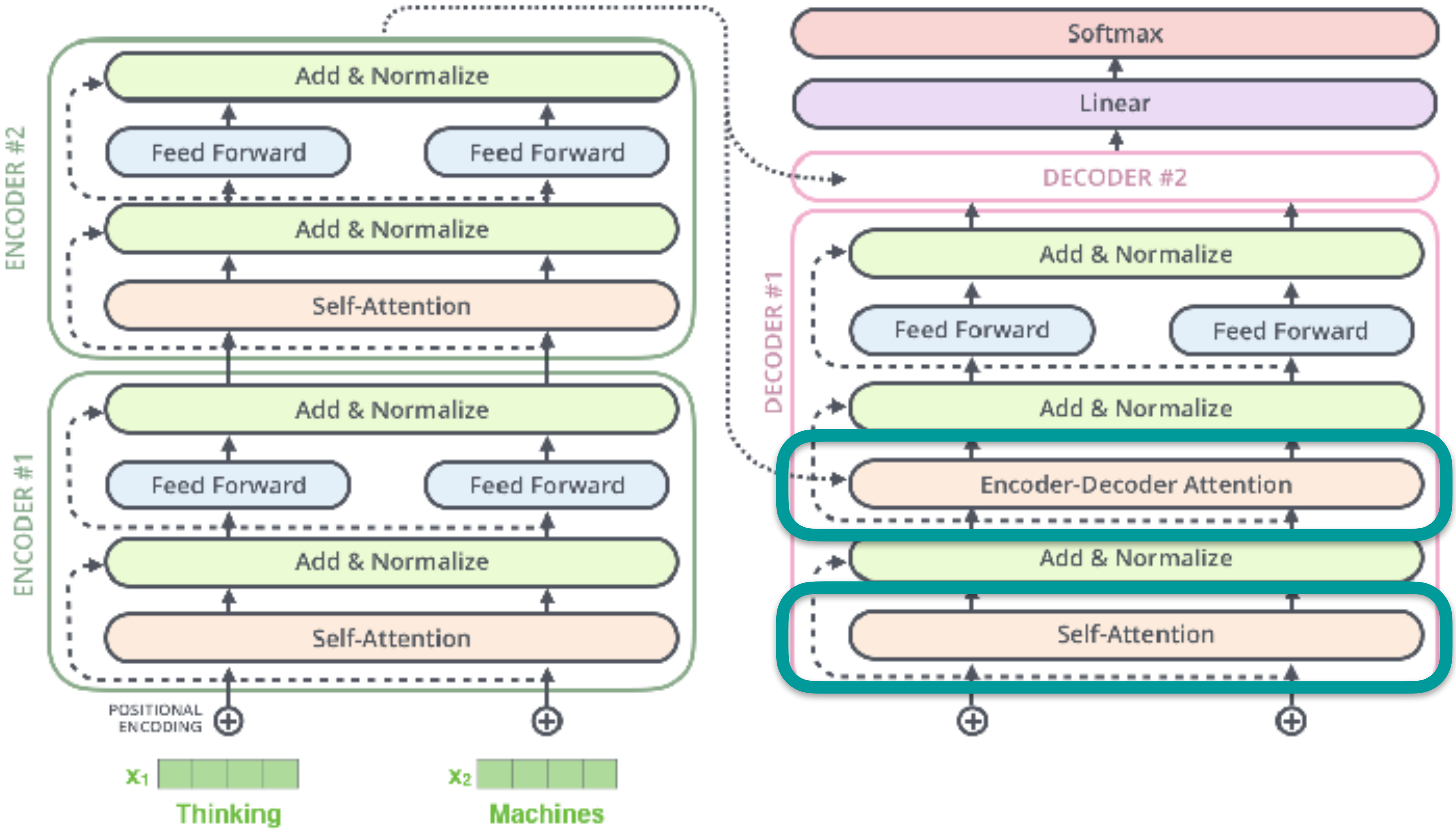
Michael Townsen Hicks[1] · James Humphries[1] · Joe Slater[1]

© The Author(s) 2024

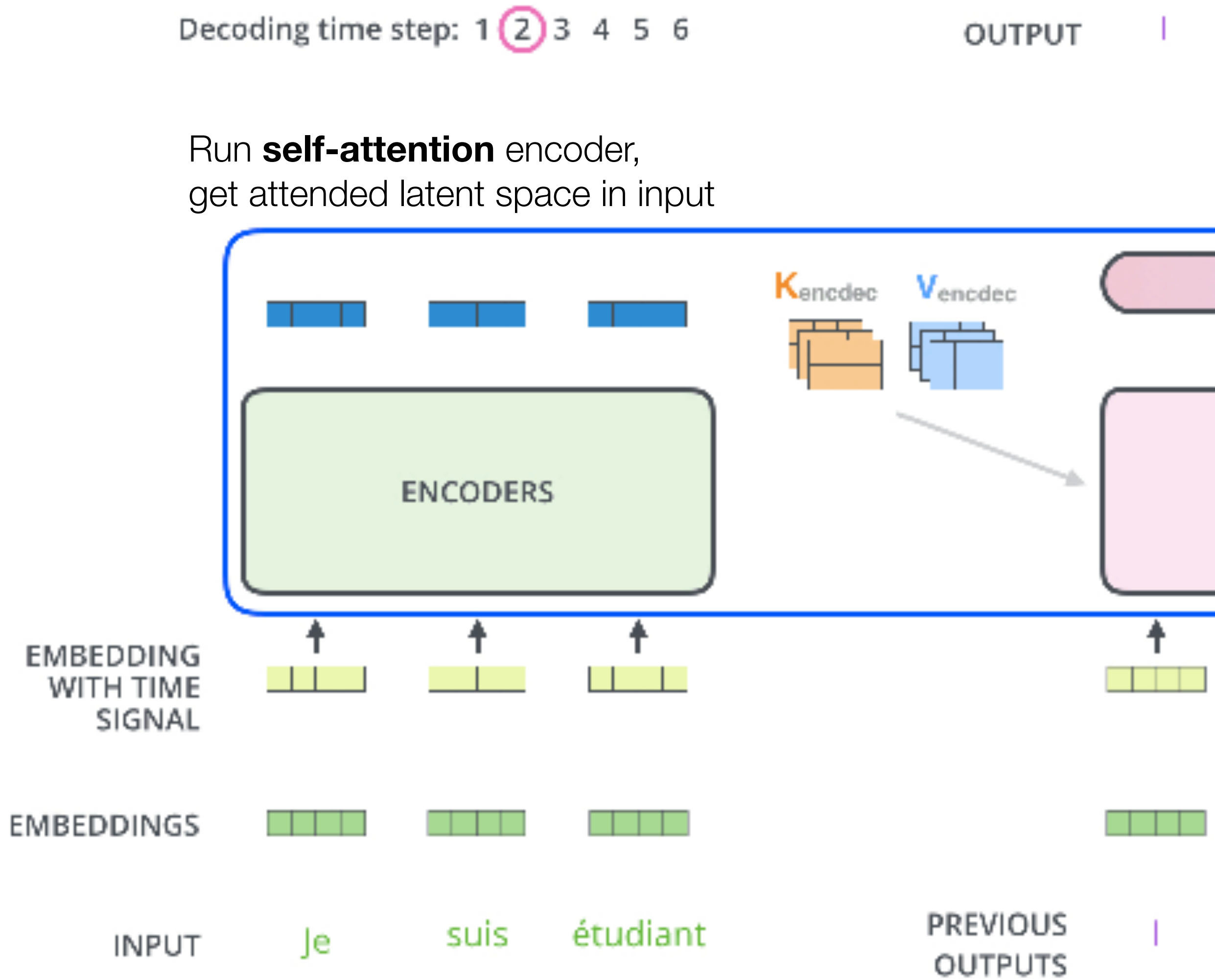
Abstract
Recently, there has been considerable interest in large language models: machine learning systems which produce human-like text and dialogue. Applications of these systems have been plagued by persistent inaccuracies in their output; these are often called "AI hallucinations". We argue that these falsehoods, and the overall activity of large language models, is better understood as *bullshit* in the sense explored by Frankfurt (On Bullshit, Princeton, 2005): the models are in an important way indifferent to the truth of their outputs. We distinguish two ways in which the models can be said to be bullshitters, and argue that they clearly meet at least one of these definitions. We further argue that describing AI misrepresentations as bullshit is both a more useful and more accurate way of predicting and discussing the behaviour of these systems.

Keywords Artificial intelligence · Large language models · LLMs · ChatGPT · Bullshit · Frankfurt · Assertion · Content

Encoders and Decoders



Encoders and Decoders



Run **self-attention** encoder,
get attended latent space in input

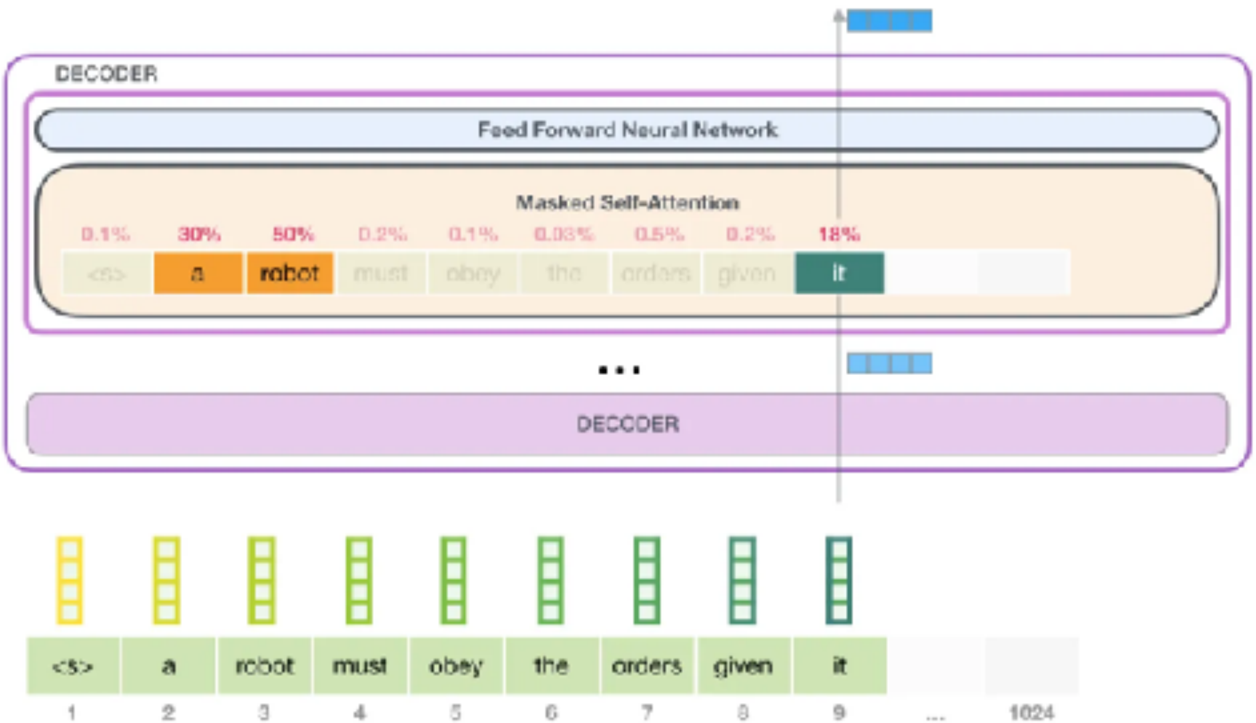
Run decoder with **cross attention** of encoder embeddings,
run in auto regressive manner

Auto-regressive Transformer, Decode Only

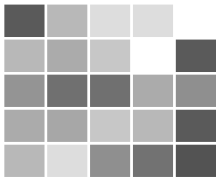
- Decoder only, text encoding happens in attention

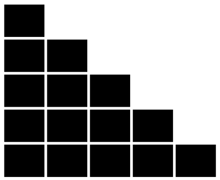
$$\mathcal{L}_1(\mathcal{U}) = \sum_{i \in S} \log P(\underbrace{u_i}_{\text{curr}} \mid \underbrace{u_{i-k}, \ldots, u_{i-1}}_{\text{other words}} ; \underbrace{\mathbf{W}}_{\text{params}})$$

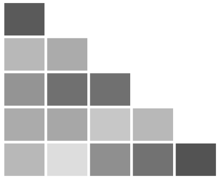
Predict the next word from unlabeled dataset



Also known as generative pre-training (GPT)


Attention Map


Mask

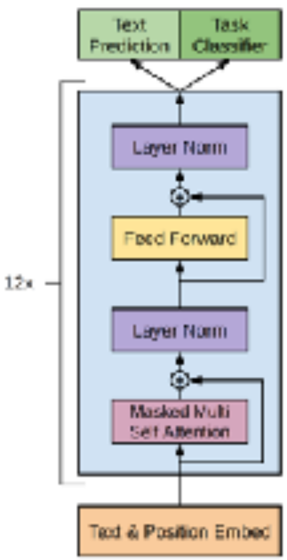

Masked Attention Map

Radford, et al. Improving Language Understanding by Generative Pre-Training, ArXiV 20

Fine Tuning after Generative Pre-training

- Supervised tasks after pre-training, make transformer better through various tasks, trained simultaneously

$\sigma(h_l^m \cdot \mathbf{W}_c + \mathbf{b}_c)$ new layer output for transfer learning



$$\mathscr{L}_2(C) = \sum_{c \in C} CE\left(y_c, \sigma(h_l^m \cdot \mathbf{W}_c + \mathbf{b}_c)\right)$$

label prediction

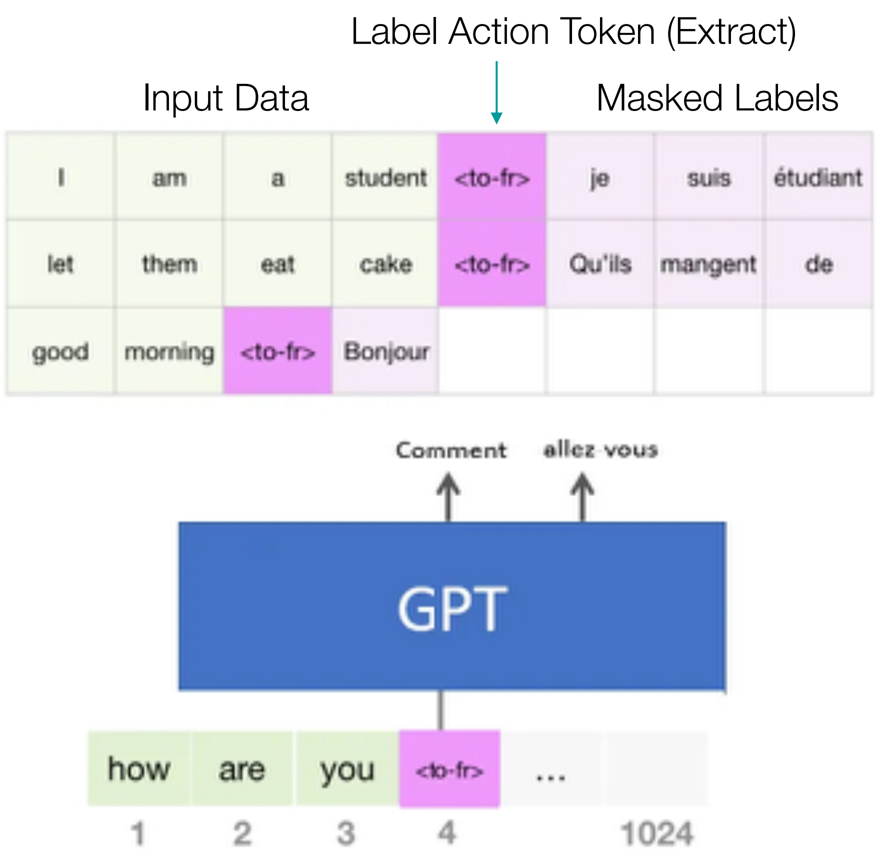
For each task, C

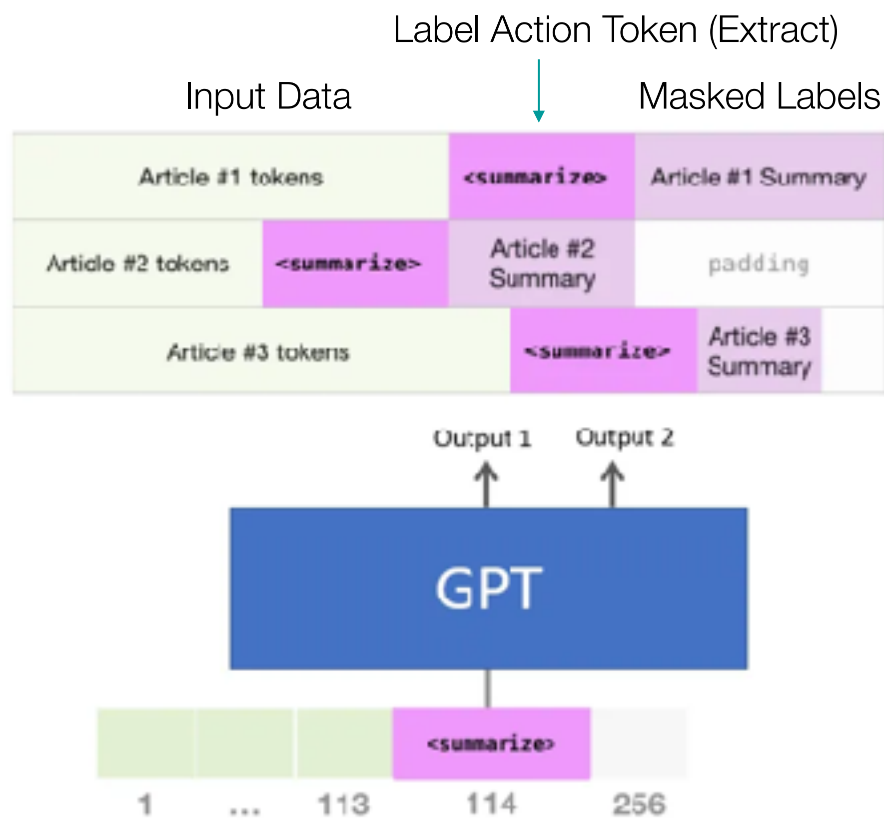
$$\mathscr{L}_{Total} = \sum_{C \in \mathscr{C}} \mathscr{L}_2(C) + \lambda \cdot \mathscr{L}_1(C)$$

prediction next word (same as GPT)

Radford, et al. Improving Language Understanding by Generative Pre-Training, ArXiV 20

Providing labels to a decoder-only model





Fine Tune with “Extract” Tokens and training examples.

Improving LLMs with Reinforcement Learning

- Generative pre-training allows for text generation in many different applications
- Some tasks it is not trained for, it can still generate plausible outcomes (called **emergent abilities**)
  - Many arguments exist for this being a result of natural language as input, which generalizes well to unseen classification tasks
- To further fine tune and make answers more relevant, reinforcement learning is often used:
  - **Game systems**: known optimal outputs, like playing chess, can be rewarded systematically
  - **Human feedback**: various tasks (like dialogue) with rewards for relevant responses and punishment for poor responses. ***Used for Chat.***
- **ChatGPT**: uses various proximal policy networks (agents) that guide next generation based on current state, needs periodic human supervision
- **DeepSeek**: mitigates human supervision in RL, opting for group sampling that balances reference distribution divergence.
  - One Insight: It helps to have many chain-of-thought examples in circulation

More Visual Sizing

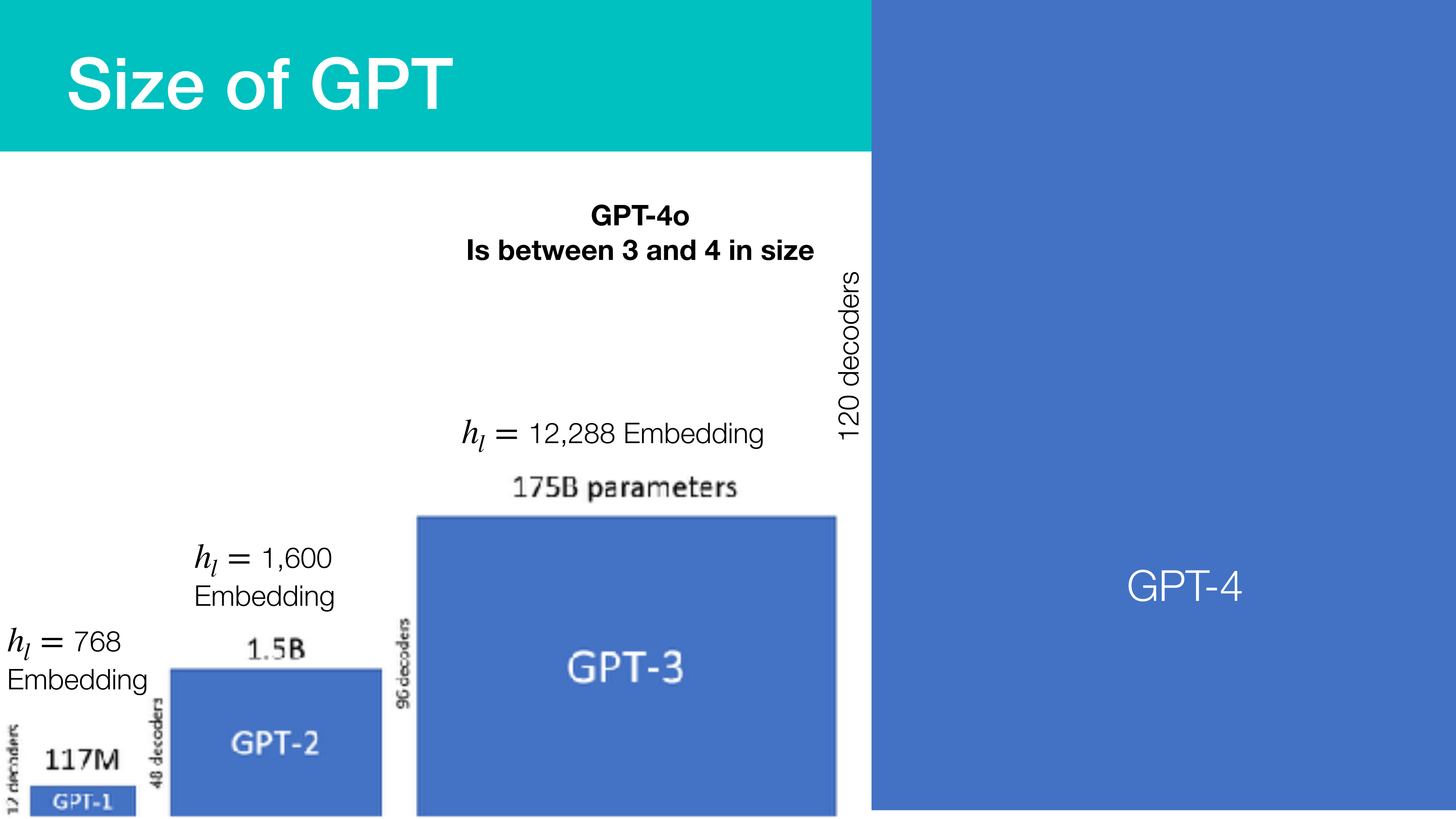
Size of GPT

GPT-4o
Is between 3 and 4 in size

h_l = 12,288 Embedding

h_l = 1,600 Embedding

h_l = 768 Embedding



512 Context Length

1024 Context Length

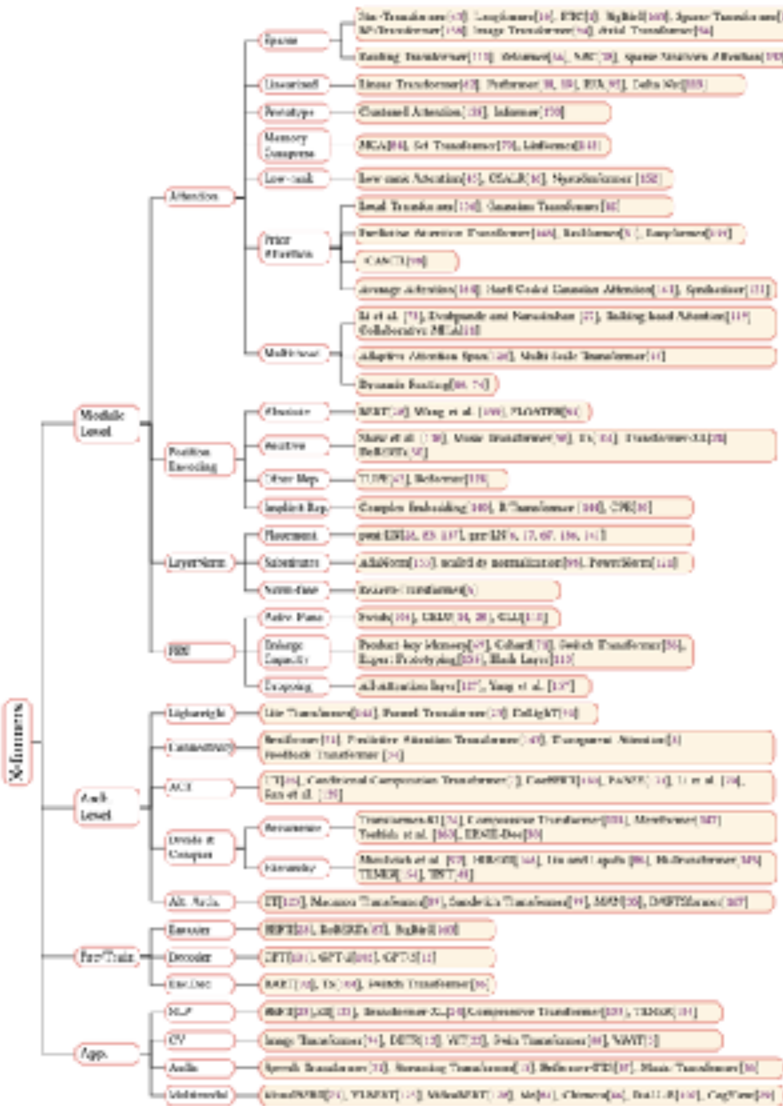
2048 Context Length

8000 (+) Context Length
1.7 Trillion parameters
h_l = 20,000 Neurons embedding
Input Images

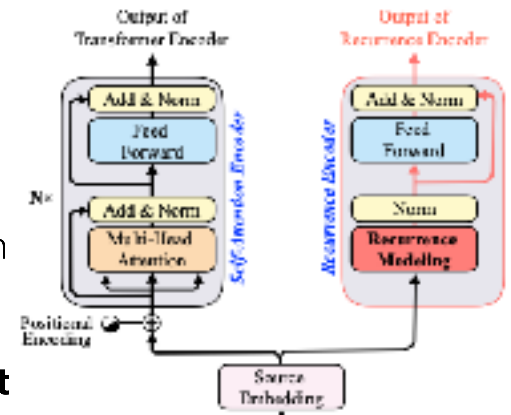
On par with BERT

https://medium.com/@YanAIx/step-by-step-into-gpt-70bc4a5d8714

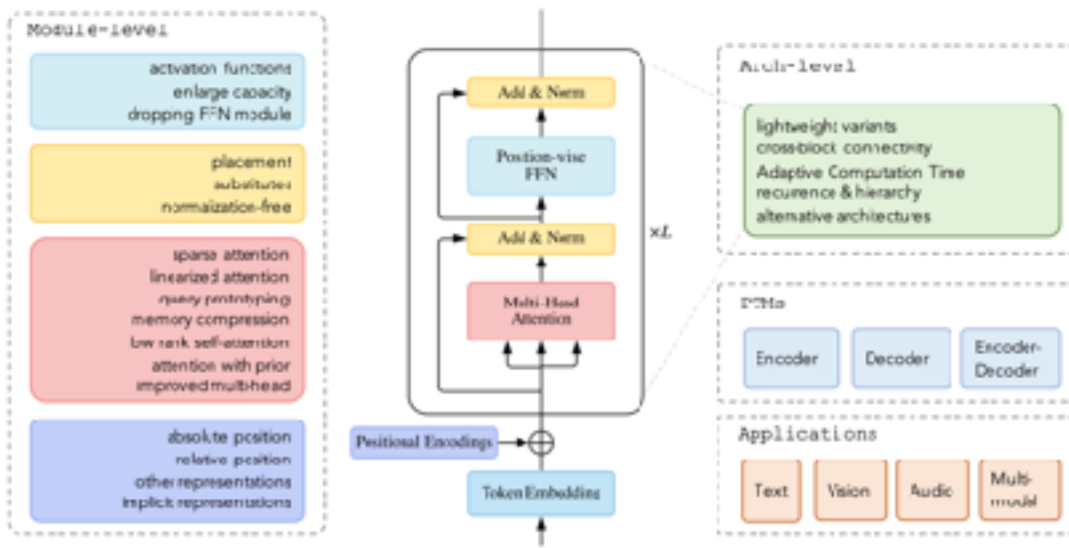
We only have skimmed the surface…



Architecture Tuning Matters
Pre-Training Matters
Sparse Attention Helps with length
Positional Encoding Doesn't
Recurrence Might… ?
X-formers are NOT just for Text



Hao et al. 2019



Lin et al "Survey of X-formers, 2021, https://arxiv.org/pdf/2106.04554.pdf

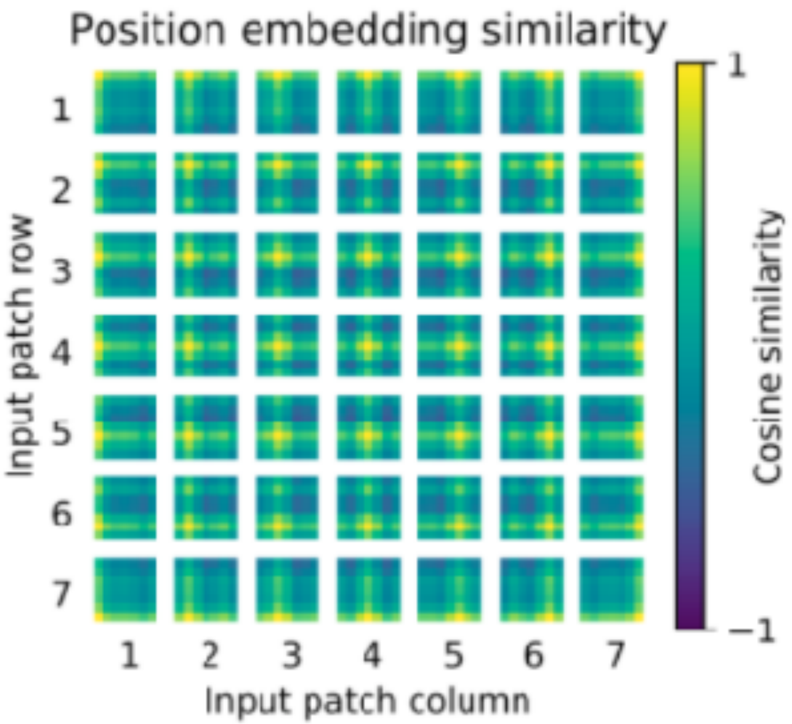
Vision Transformers

Vision Transformers Video

https://ai.googleblog.com/2020/12/transformers-for-image-recognition-at.html?m=1

Vision Transformers

- Divide image into patches
  - Treat each patch as something to encode separately
  - Flatten each patch
  - Put through dense layer
- Add positional encoding based on position of patch
  - for 7x7 patch, there are 49 positions
- Put into transformer. Same as text transformers …
- **But you need a lot of data**
  - 14M or more images seems to be sweet spot



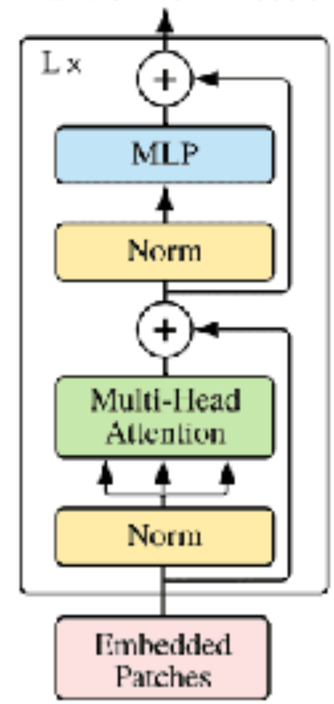
ViT Architectures Parameters

- *D* is size of patch embedding
- Uses skip connections (all size *D*)
- Multi-headed self attention (MSA) takes *D* input `patch_embed + pos_embed`
- Three sized architectures differ in:
  - *L* blocks used (*i.e.*, "layers")
  - *H* heads in each layer (*i.e.*, "heads")
  - MLP head is final classifier

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos},$$
$$\mathbf{z}'_\ell = \text{MSA}(\text{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1},$$
$$\mathbf{z}_\ell = \text{MLP}(\text{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell,$$
$$\mathbf{y} = \text{LN}(\mathbf{z}_L^0)$$

Self attention

Transformer Encoder



Model	Layers	Hidden size *D*	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

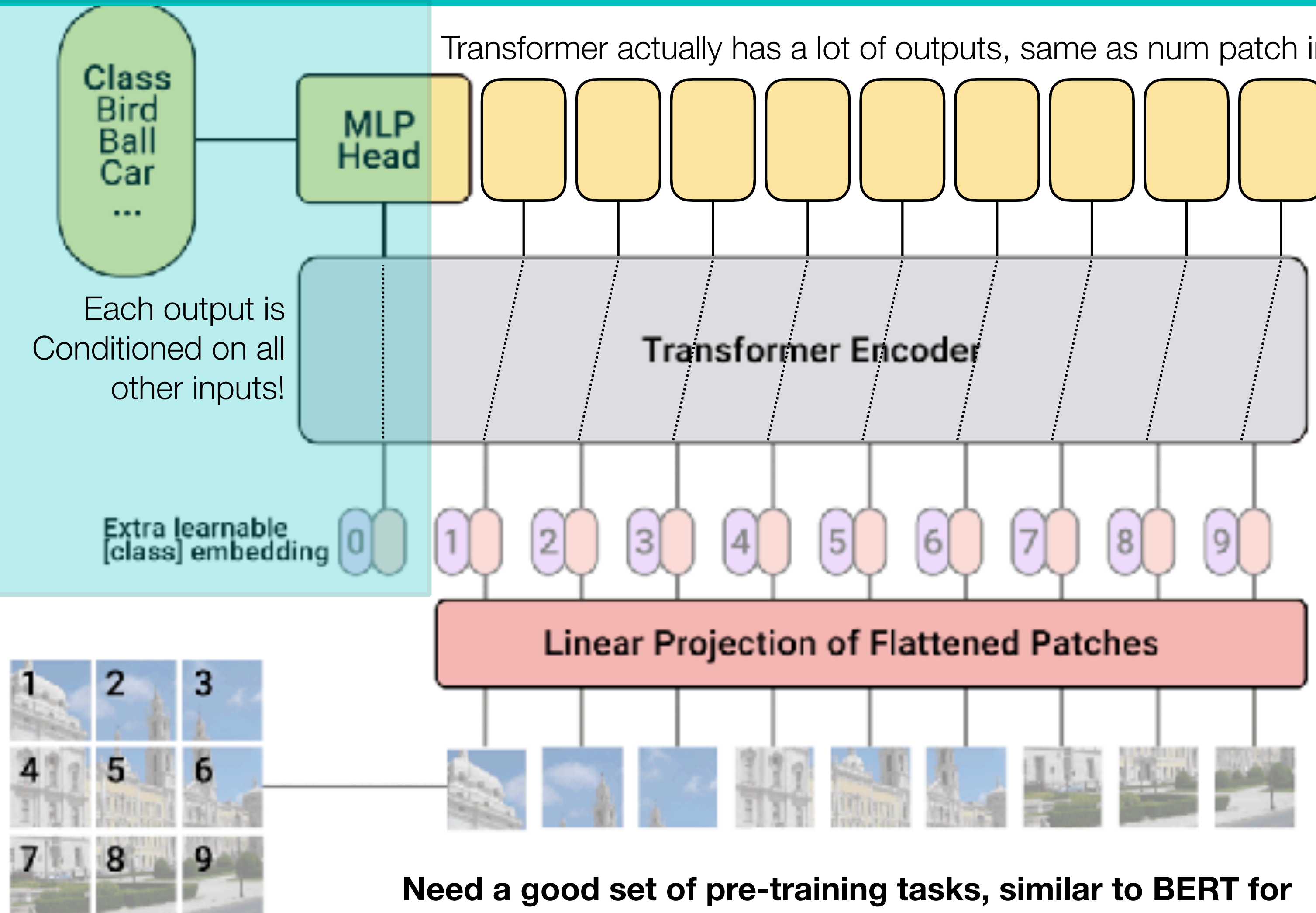
ResNet50: 23M

Dosovitskiy *et al.* (2021) An image is worth 16x16 words: Transformers for Image Recognition at Scale https://arxiv.org/abs/2010.11929

Learnable class embedding

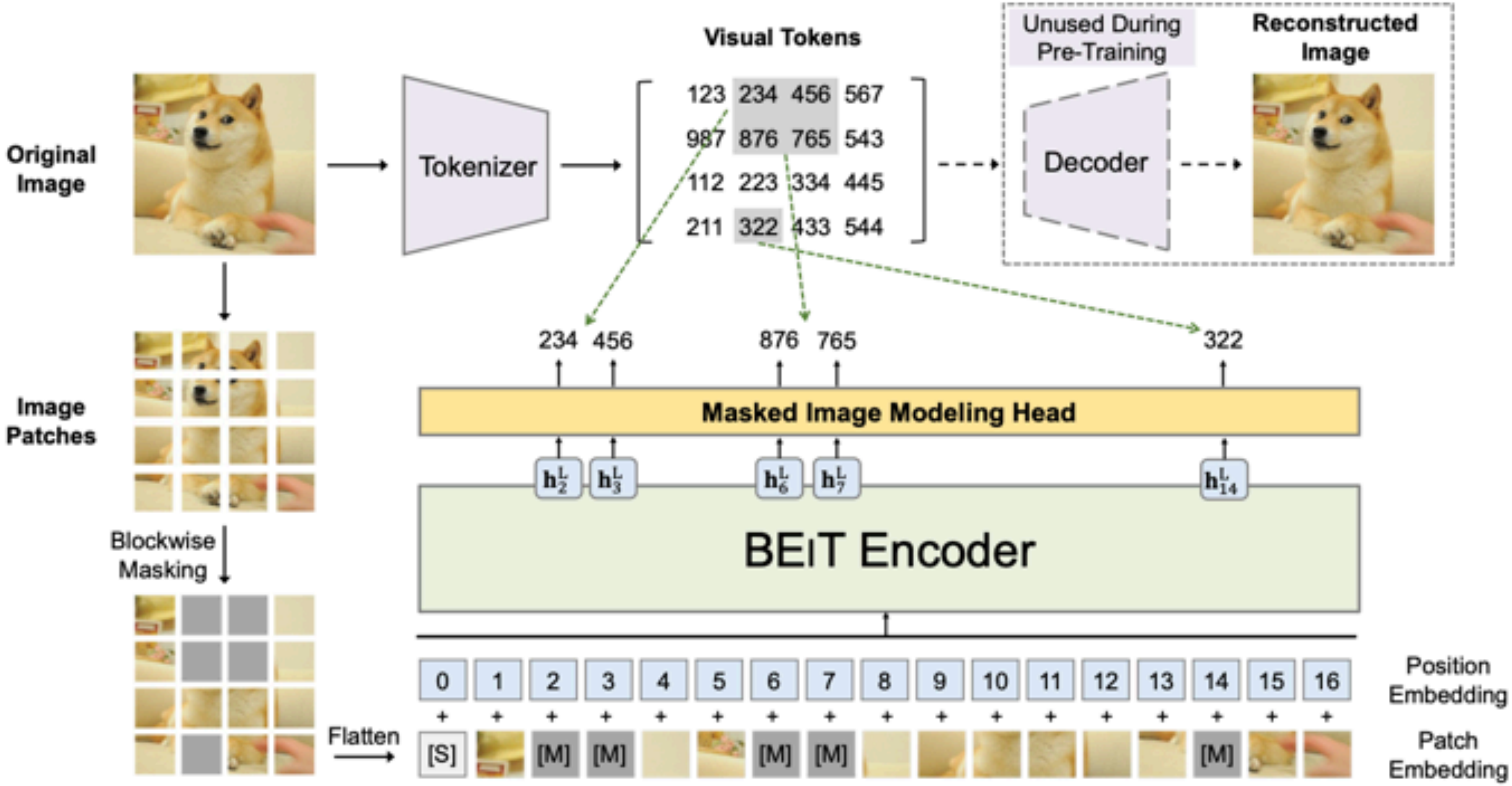
Transformer actually has a lot of outputs, same as num patch inputs

Each output is Conditioned on all other inputs!



Need a good set of pre-training tasks, similar to BERT for
(1) CLS token training
(2) Each patch output

Pre-training for ViT

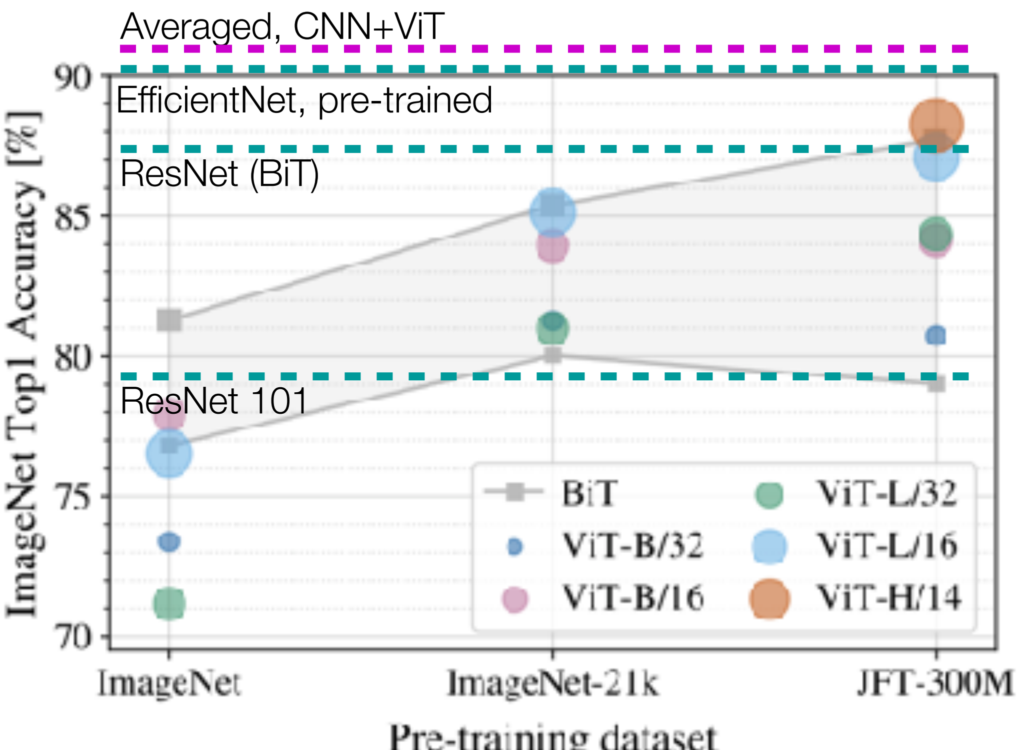


Downstream Fine Tuning: Image classification [CLS], semantic segmentation (patch output), and others…

https://www.fast.ai/2020/01/13/self_supervised/

Fine tuning: Do they work?

- Less than 14M images for pre-training? Do not use as a base model!
    - CNNs will be easier and very performant…



~1M Images
1k classes

~14M Images (part. crawl)
21k classes

~300M Images (crawled)
18k classes

Transfer Learning From Huge ViT

	BeIT	ViT-H		Previous SOTA
ImageNet	89.5	88.55	←	88.5
ImageNet-ReaL		90.72	←	90.55
Cifar-10		99.50	←	99.37
Cifar-100	91.8	94.55	←	93.51
Pets		97.56	←	96.62
Flowers		99.68	←	99.63