Deep Learning Crash Course

Hui Xue

Fall 2021

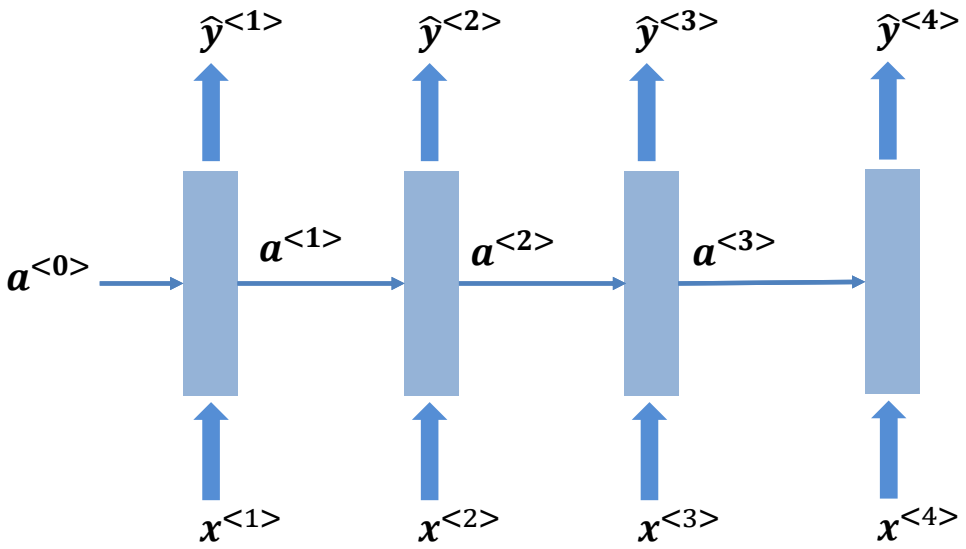
www.deeplearningcrashcourse.org

Outline

- Attention mechanism in detail
- Seq2Seq Transformer
- Review on Elmo, Bert, GPT models
- Trending towards bigger model
- Attention for images

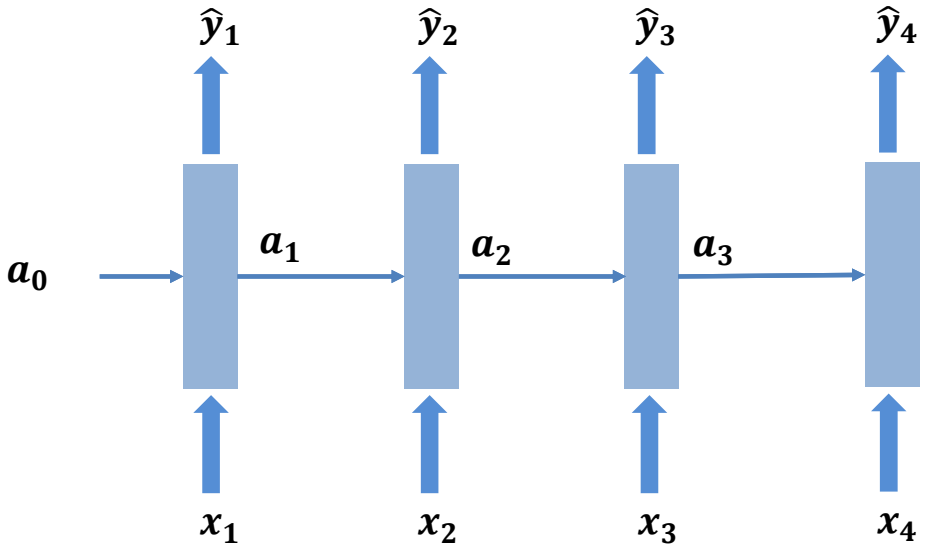
https://www.cnet.com/news/transformers-animated-movie-in-production-despite-coronavirus-quarantine/

Modeling sequence with variable length



- RNN or bidirectional RNN
- Sequential processing
- Use internal state to carry information through time

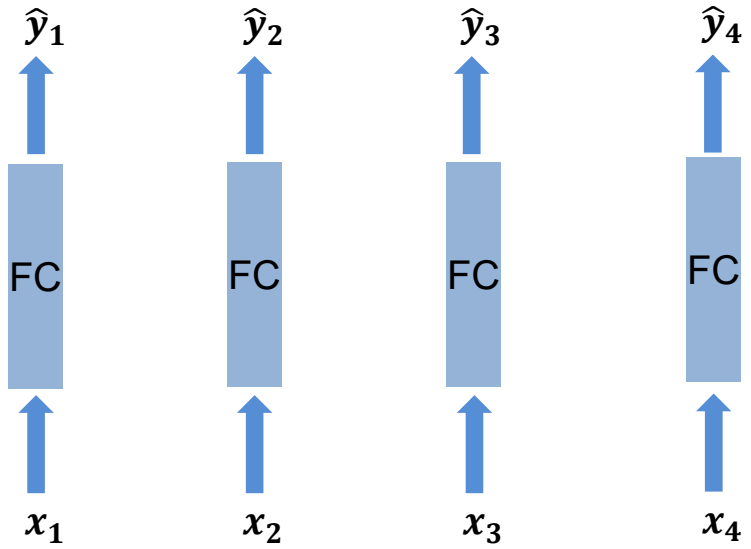
Modeling sequence with variable length



Change the notation for time steps

- RNN or bidirectional RNN
- Sequential processing
- Use internal state to carry information through time

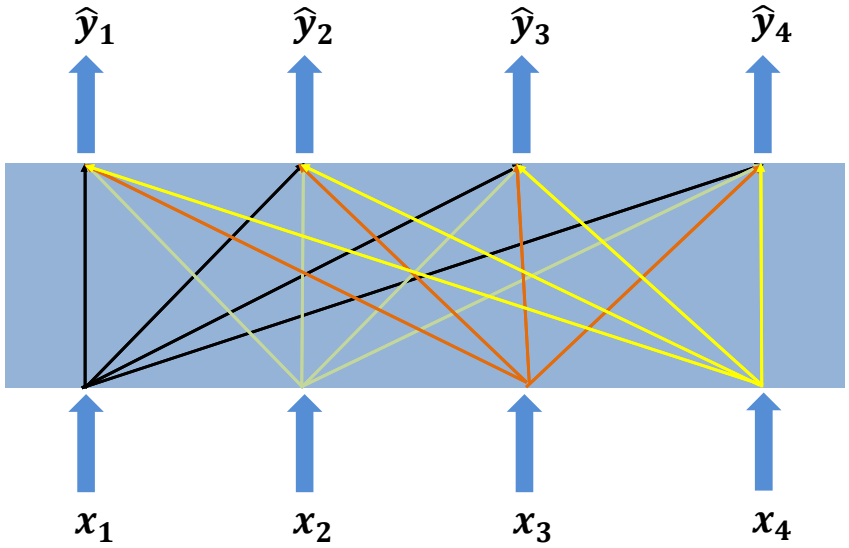
Best of two worlds



The ideal model should:

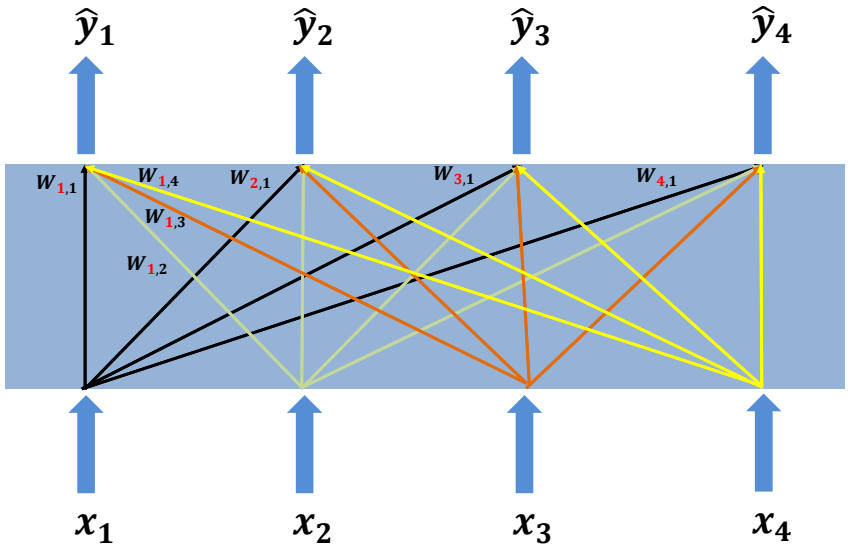
- Parallel processing
- Handle sequence data with variable length

Best of two worlds: self-attention



- Parallel processing
- Handle sequence data with variable length
- Every output takes contributions from all inputs

Self-attention

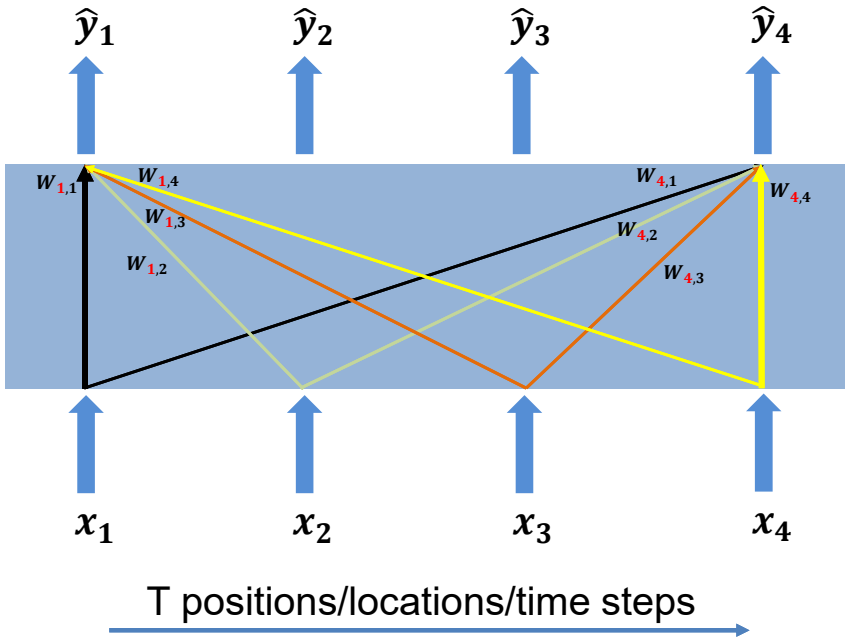


Let's compute output at location i as the linear combination of all inputs

$$\hat{y}_i = \sum_{j=1}^{T} \alpha_{i,j} x_j$$

Need an intuitive way is to compute weights $\alpha_{i,j}$

Self-attention



How to determine the weights?

- Measure how "close" of two vectors x_i and x_j

$$\alpha_{i,j} = x_i^T x_j$$

One choice is the inner product

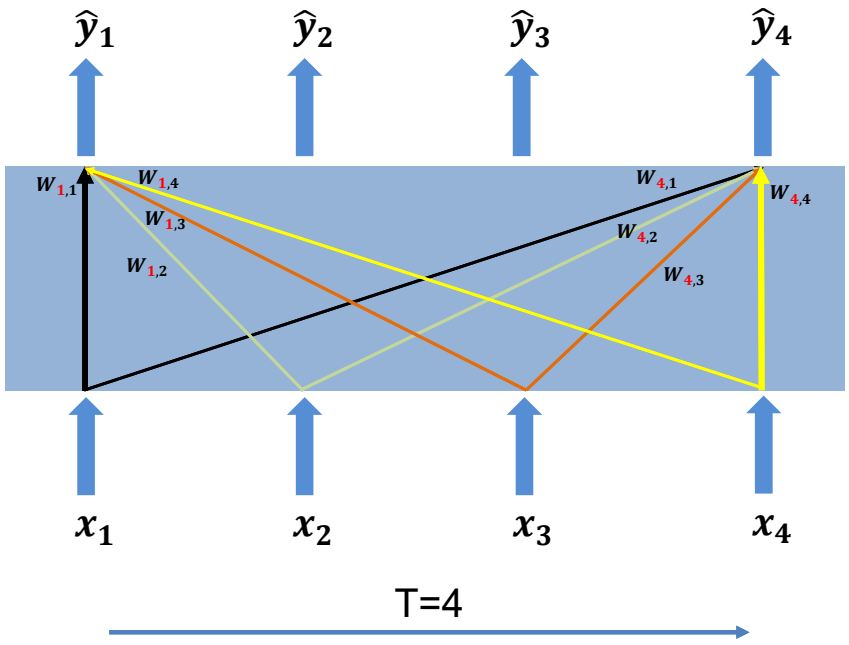
If all x vectors have similar magnitudes e.g., $\alpha_{i,i}$ often is the most significant

x_i: Nx1 vector

$X = [x_1 \quad x_2 \quad x_3 \quad x_4]$, NxT matrix

$W = [\alpha_{i,j}]$, TxT matrix

Self-attention



x_i: Nx1 vector

$X = [x_1 \quad x_2 \quad x_3 \quad x_4]$, Nx4 matrix

$W = [\alpha_{i,j}]$, 4x4 matrix

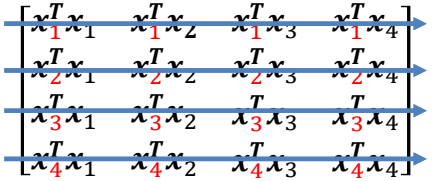
$$\widehat{W} = X^T X = \begin{bmatrix} x_1^T \\ x_2^T \\ x_3^T \\ x_4^T \end{bmatrix} [x_1 \quad x_2 \quad x_3 \quad x_4]$$

$$= \begin{bmatrix} x_1^T x_1 & x_1^T x_2 & x_1^T x_3 & x_1^T x_4 \\ x_2^T x_1 & x_2^T x_2 & x_2^T x_3 & x_2^T x_4 \\ x_3^T x_1 & x_3^T x_2 & x_3^T x_3 & x_3^T x_4 \\ x_4^T x_1 & x_4^T x_2 & x_4^T x_3 & x_4^T x_4 \end{bmatrix}$$

Self-attention

$$\widehat{W} = X^T X = \begin{bmatrix} x_1^T x_1 & x_1^T x_2 & x_1^T x_3 & x_1^T x_4 \\ x_2^T x_1 & x_2^T x_2 & x_2^T x_3 & x_2^T x_4 \\ x_3^T x_1 & x_3^T x_2 & x_3^T x_3 & x_3^T x_4 \\ x_4^T x_1 & x_4^T x_2 & x_4^T x_3 & x_4^T x_4 \end{bmatrix}$$

$$W = softmax(\widehat{W}, dim = 1)$$



Softmax along row

Attention score

$$W = \begin{bmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \alpha_{1,4} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} & \alpha_{2,4} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & \alpha_{3,4} \\ \alpha_{4,1} & \alpha_{4,2} & \alpha_{4,3} & \alpha_{4,4} \end{bmatrix}$$

$$Y = [\hat{y}_1 \quad \hat{y}_2 \quad \hat{y}_3 \quad \hat{y}_4], \text{ Nx4 matrix}$$

$$Y = XW^T$$

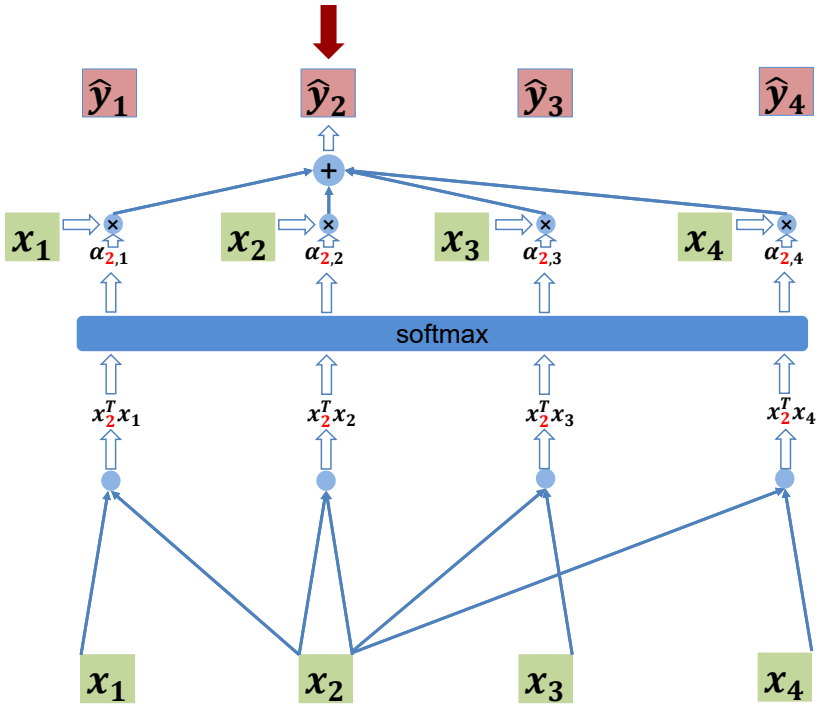
$$= [x_1 \quad x_2 \quad x_3 \quad x_4] \begin{bmatrix} \alpha_{1,1} & \alpha_{2,1} & \alpha_{3,1} & \alpha_{4,1} \\ \alpha_{1,2} & \alpha_{2,2} & \alpha_{3,2} & \alpha_{4,2} \\ \alpha_{1,3} & \alpha_{2,3} & \alpha_{3,3} & \alpha_{4,3} \\ \alpha_{1,4} & \alpha_{2,4} & \alpha_{3,4} & \alpha_{4,4} \end{bmatrix}$$

$$\alpha_{i,j} = \frac{exp(x_i^T x_j)}{\sum_{k=1}^{T} exp(x_i^T x_k)}$$

$$\hat{y}_1 = \alpha_{1,1} x_1 + \alpha_{1,2} x_2 + \alpha_{1,3} x_3 + \alpha_{1,4} x_4$$

Self-attention

We want to compute output at this position



x_i: Nx1 vector

$X = [x_1 \quad x_2 \quad x_3 \quad x_4]$, NxT matrix

$\widehat{W} = X^T X$, TxT matrix

$W = softmax(\widehat{W}, dim = 1) = [\alpha_{i,j}]$, TxT matrix

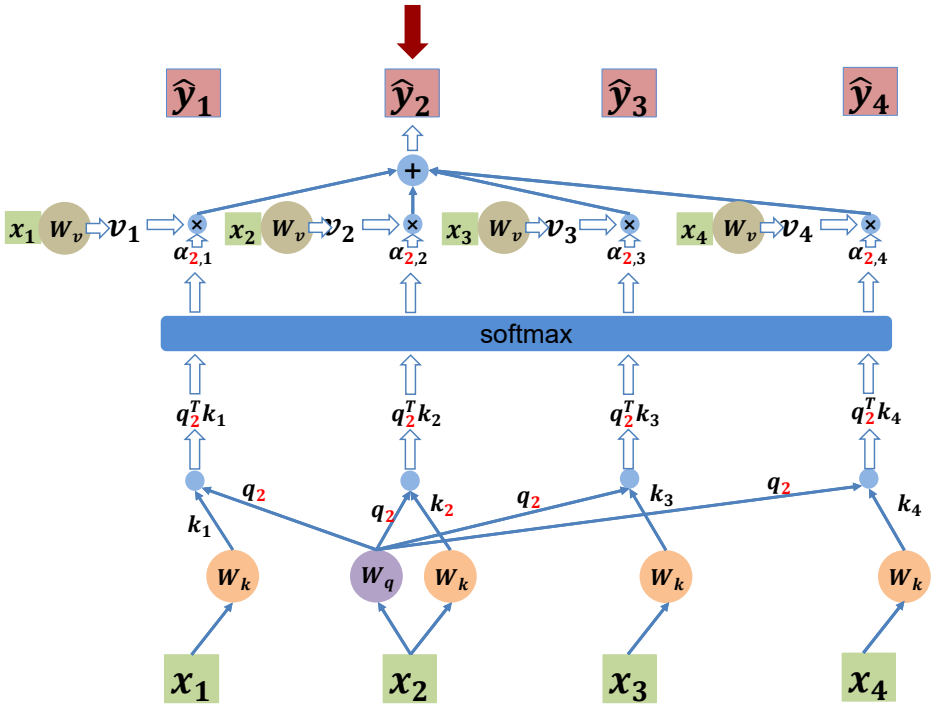
$Y = [\hat{y}_1 \quad \hat{y}_2 \quad \hat{y}_3 \quad \hat{y}_4] = XW^T$, NxT matrix

$$\hat{y}_i = \sum_{j=1}^{T} \alpha_{i,j} x_j$$

- Output is a weighted average of inputs
- Weights measure how "close" two input vectors are
- Given output position i, a x_j that is "closer" to input x_i has higher weight

Attention

We want to compute output at this position



- Self-attention
  - √ parallel computing
  - √ process sequence data
- Has no learnable parameters

So let's introduce learnable parameters

$$q_i = W_q x_i \quad \text{query}$$

$$k_i = W_k x_i \quad \text{key}$$

$$v_i = W_v x_i \quad \text{value}$$

Bias term can be added too

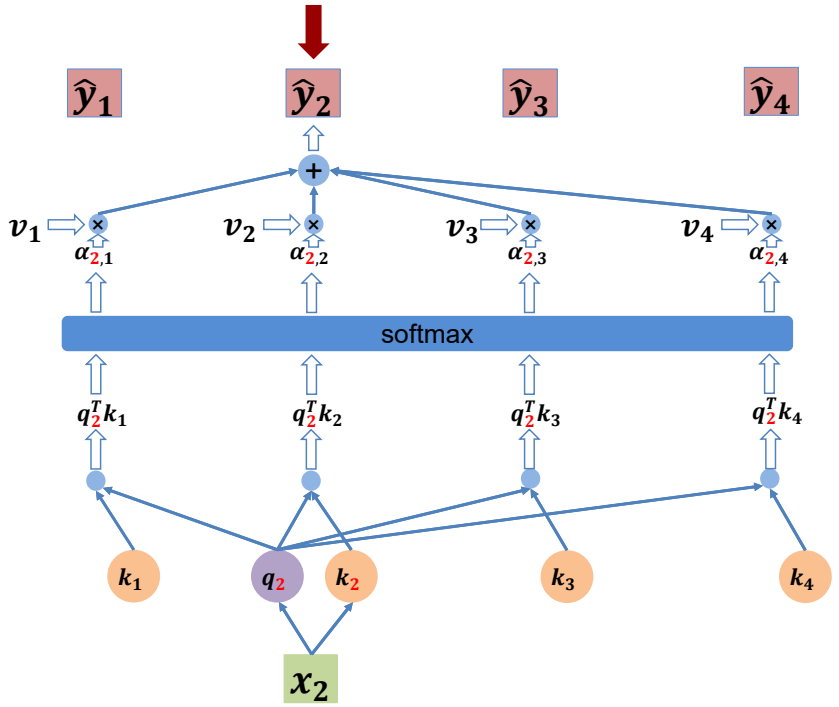
W_q and W_k : DxN matrix
W_v : VxN matrix

D is the working dimension of attention

Q and K : DxT matrix
V: VxT matrix

Attention

We want to compute output at this position



x_i: Nx1 vector

$X = [x_1 \quad x_2 \quad x_3 \quad x_4]$, NxT matrix

W_q and W_k : DxN matrix; W_v: VxN matrix

$Q = W_q X$, DxT matrix
$K = W_k X$, DxT matrix
$V = W_v X$, VxT matrix

Learnable parameters:
W_q, W_k, W_v

Do not depend on T

$\widehat{W} = Q^T K$, TxT matrix

$W = softmax(\widehat{W}, dim = 1) = [\alpha_{i,j}]$, TxT matrix

$Y = VW^T$, VxT matrix

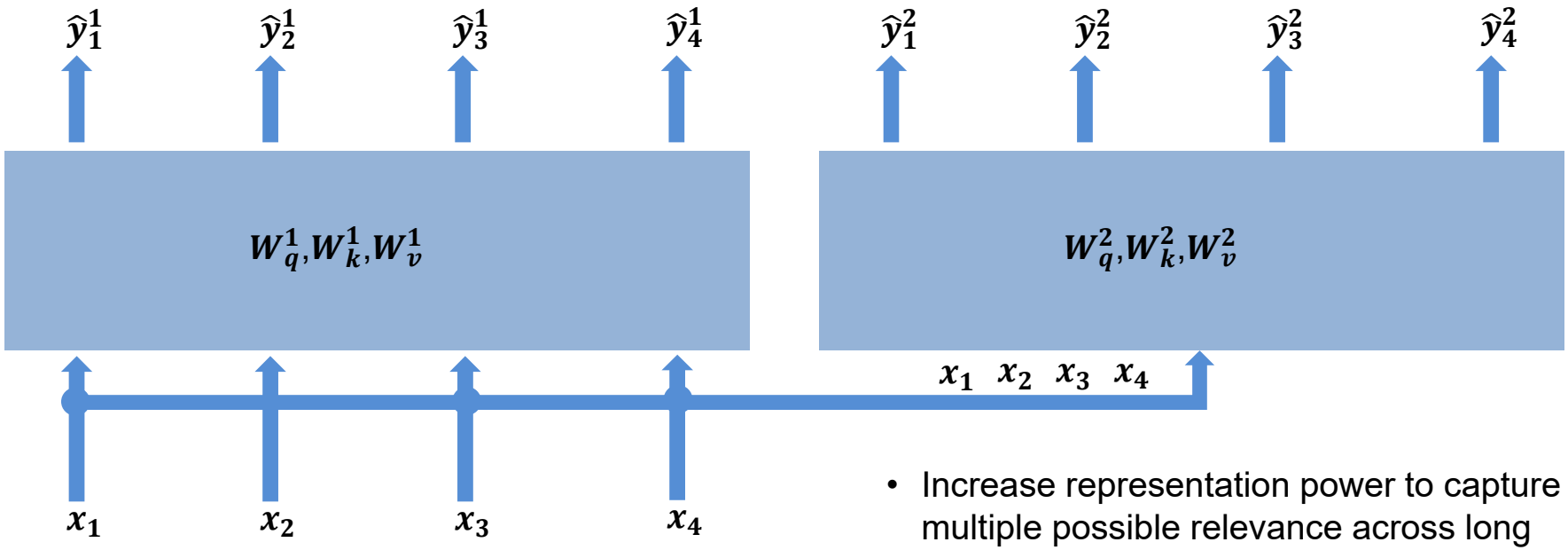
Scaled Attention

$$\alpha_{i,j} = \boldsymbol{q}_i^T \boldsymbol{k}_j \quad \Longrightarrow \quad \alpha_{i,j} = \frac{\boldsymbol{q}_i^T \boldsymbol{k}_j}{\sqrt{\boldsymbol{D}}}$$

$\widehat{\boldsymbol{W}} = \frac{\boldsymbol{Q}^T \boldsymbol{K}}{\sqrt{\boldsymbol{D}}}$, TxT matrix

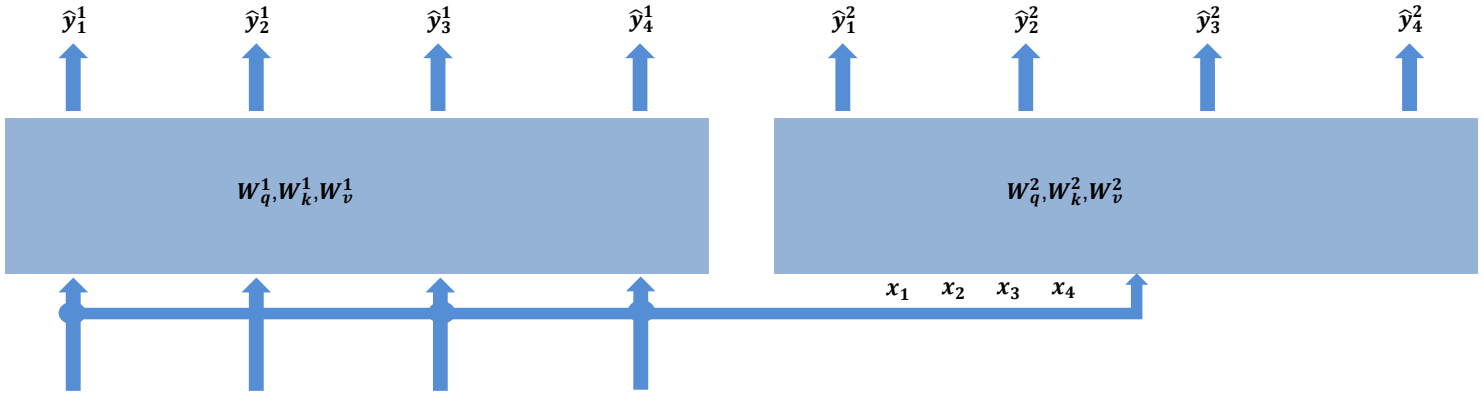
- Dot product increases with higher dimension
- A unit vector of dimension D has length $\sqrt{\boldsymbol{D}}$

Multi-head Attention



- Increase representation power to capture multiple possible relevance across long sequences

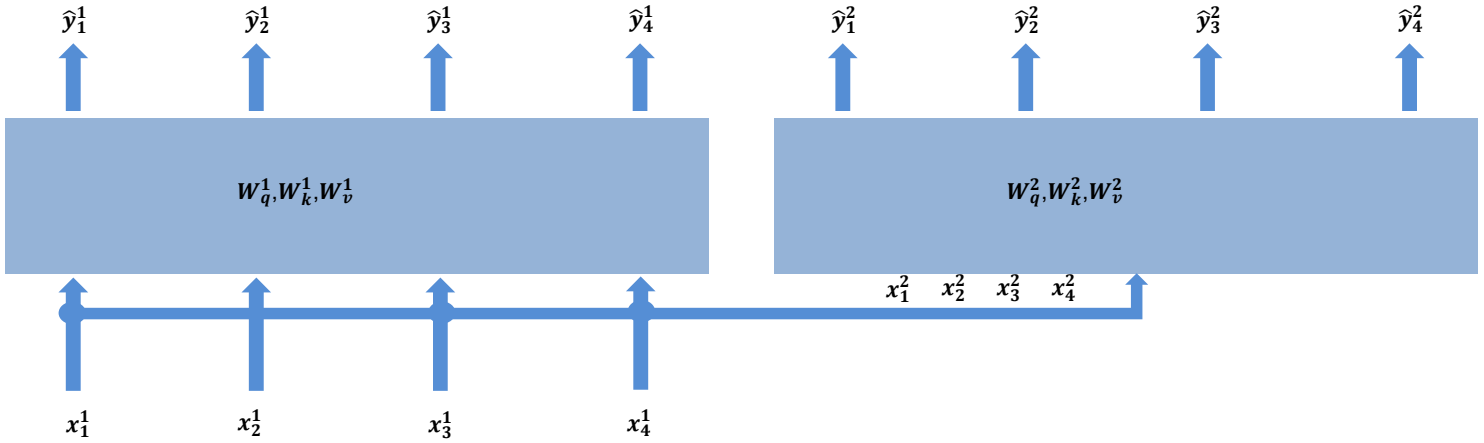
Multi-head Attention



$$\hat{y}_1 = W_o \begin{bmatrix} \hat{y}_1^1 \\ \hat{y}_1^2 \end{bmatrix}$$

W_o : Vx2V matrix, learnable parameter

Multi-head Attention



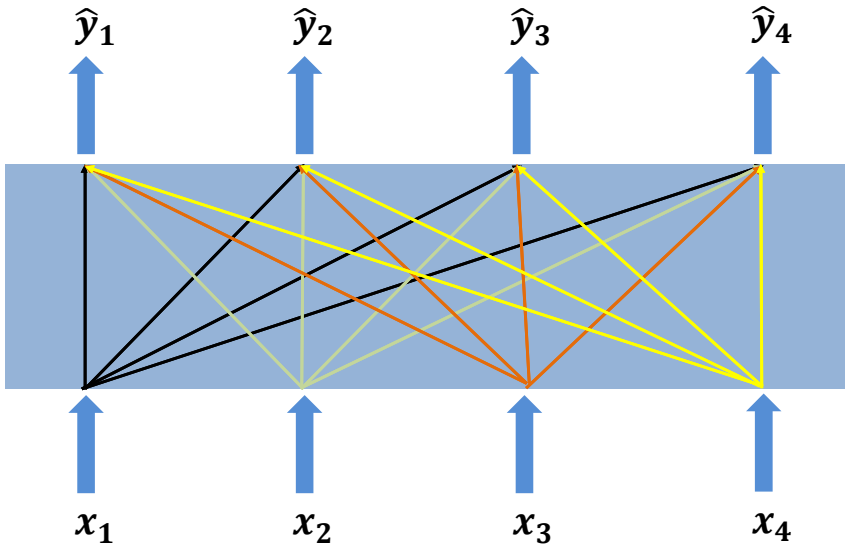
$$\hat{y}_1 = W_o \begin{bmatrix} \hat{y}_1^1 \\ \hat{y}_1^2 \end{bmatrix}$$ W_o : VxV matrix, learnable parameter

Also, possible to split input vector into multiple chunks, e.g. if H=2 for two heads,

$x_1 \Longrightarrow x_1^1 = x_1[0:N/2],\ x_1^2 = x_1[N/2:N]$ To reduce the computation

Nx1

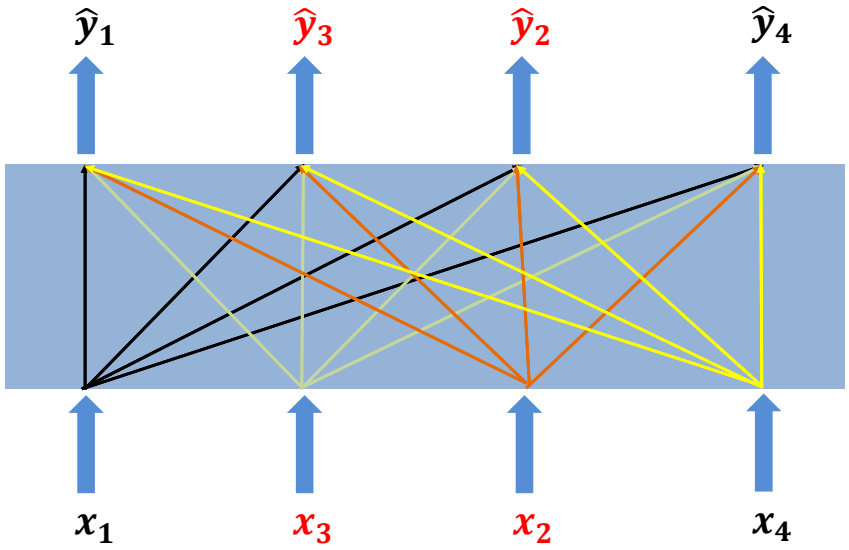
Attention is a set-to-set operation



- Computational operations for x_i and x_j are 100% symmetric
- If we switch position of x_i and x_j, outputs stay the same, but only with switched position

  if $x_i \Leftrightarrow x_j$, then $\hat{y}_i \Leftrightarrow \hat{y}_j$

Attention is a set-to-set operation



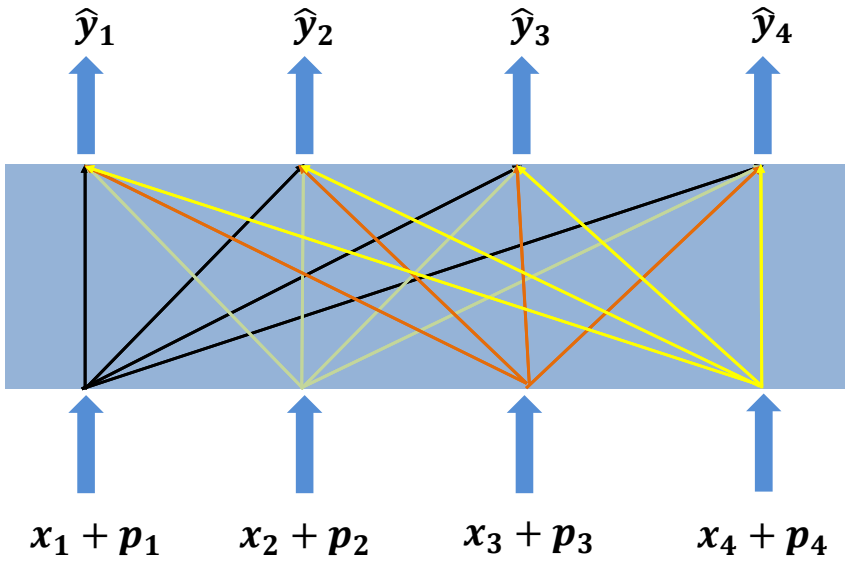
- Computational operations for x_i and x_j are 100% identical
- If we switch position of x_i and x_j, outputs stay the same, but only with switched position

  if $x_i \Leftrightarrow x_j$, then $\hat{y}_i \Leftrightarrow \hat{y}_j$

  Attention(Permute(X))=Permute(Attention(X))

- Non-causal

Add position information: Position encoding



- Position encoding

A designed, deterministic function to map position to a N dimensional vector

$$p = f(loc)$$

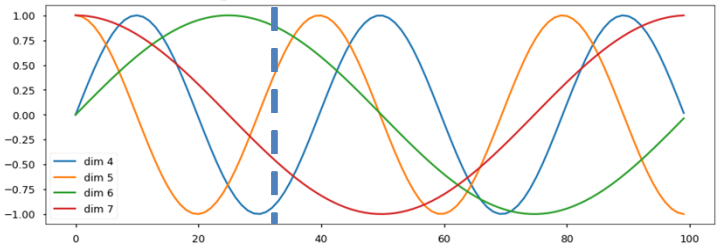
Handcrafted, but can work very well

- Add position vector to input

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

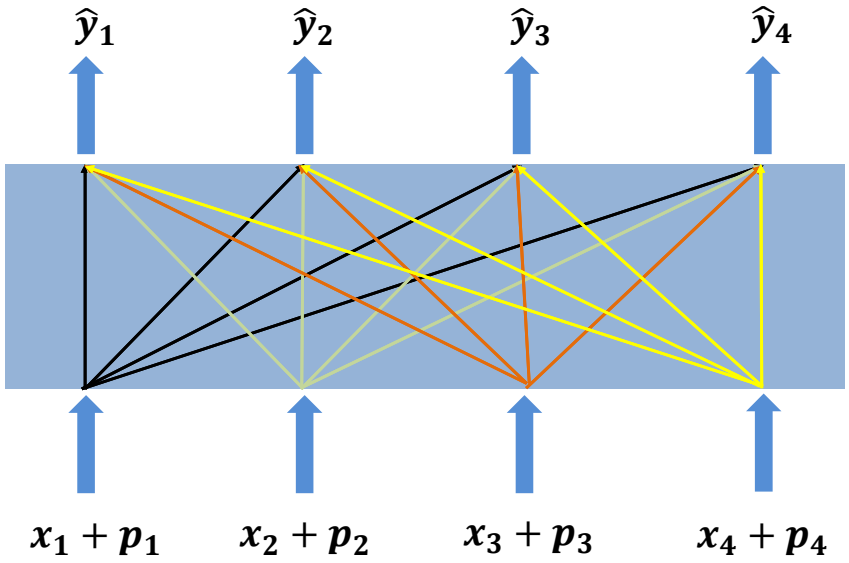
where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$. We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}.

p vector



Attention Is All You Need. https://arxiv.org/abs/1706.03762

Add position information: Position embedding



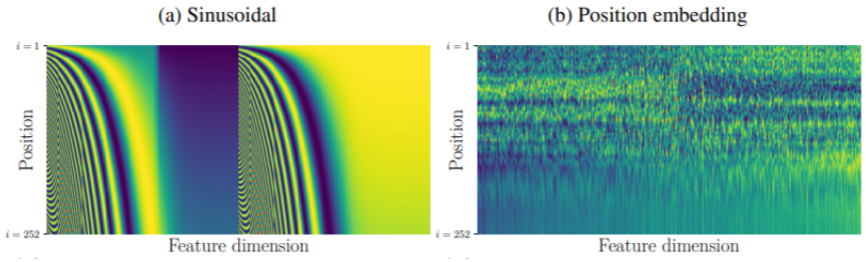
- Position embedding

$$p = f(loc)$$

Learn the mapping function f

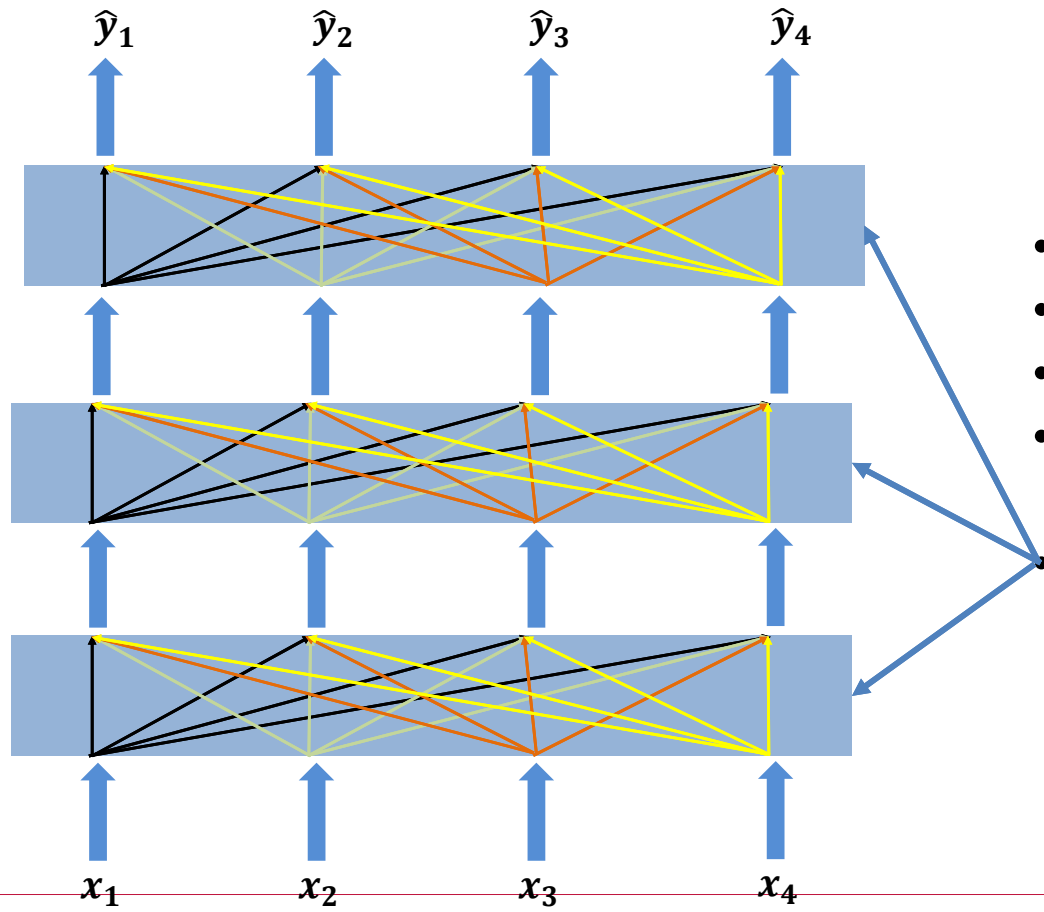
Limited by the sequence length, less flexible

- Add position vector to input



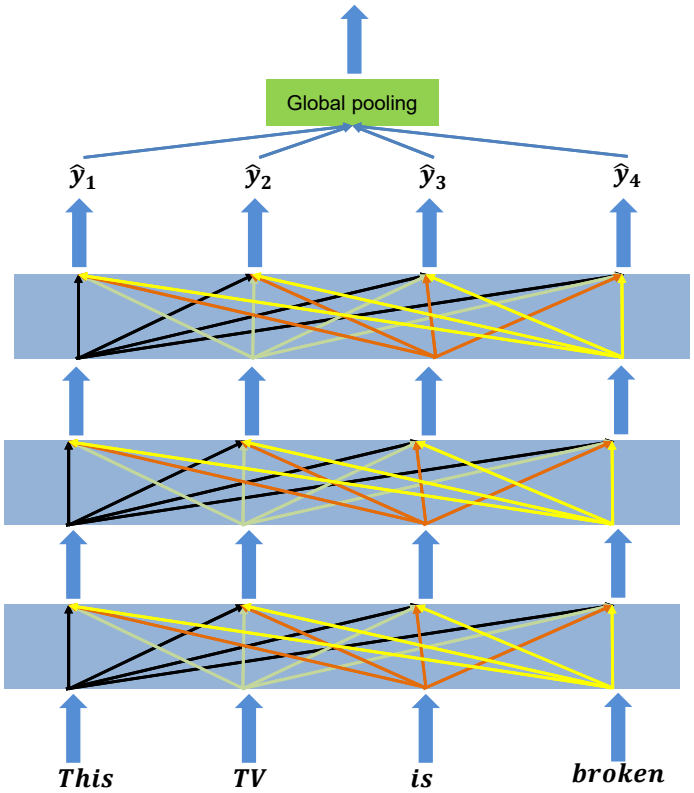
Learning to Encode Position for Transformer with Continuous Dynamical Model. https://arxiv.org/abs/2003.09229 https://towardsdatascience.com/master-positional-encoding-part-i-63c05d90a0c3

Use attention layer



- As flexible as RNN
- Non-causal by default
- Parallel computing
- Replacing RNN in many sequence models
- Attention+non-linearity+FC or CONV -> a transformer module

Use attention layer

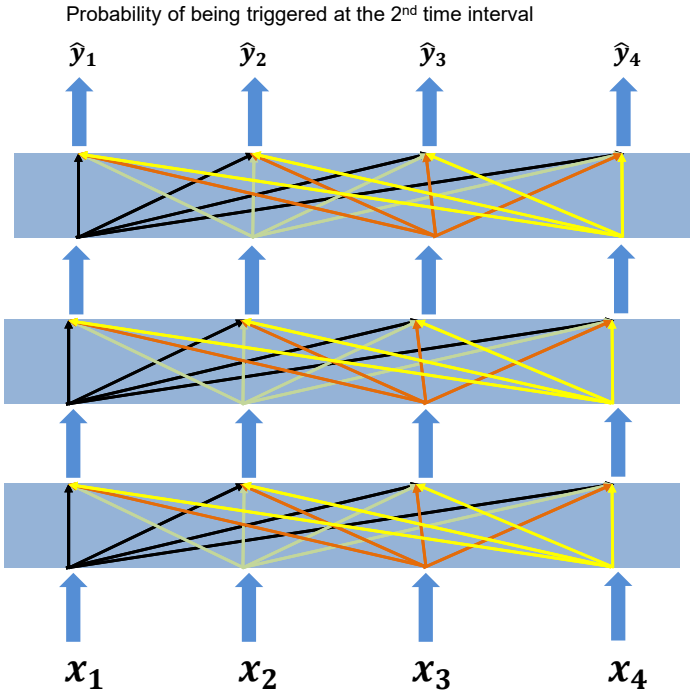


Sentimental analysis

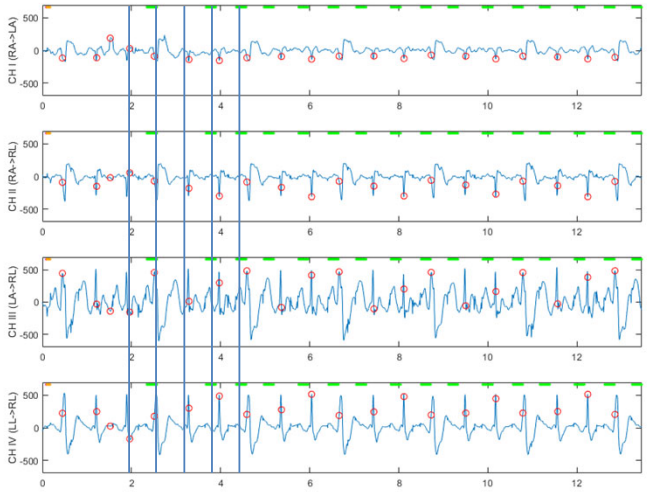
I bought this TV last week. It arrived today. This TV is broken …

Positive review or negative review?

Use attention layer

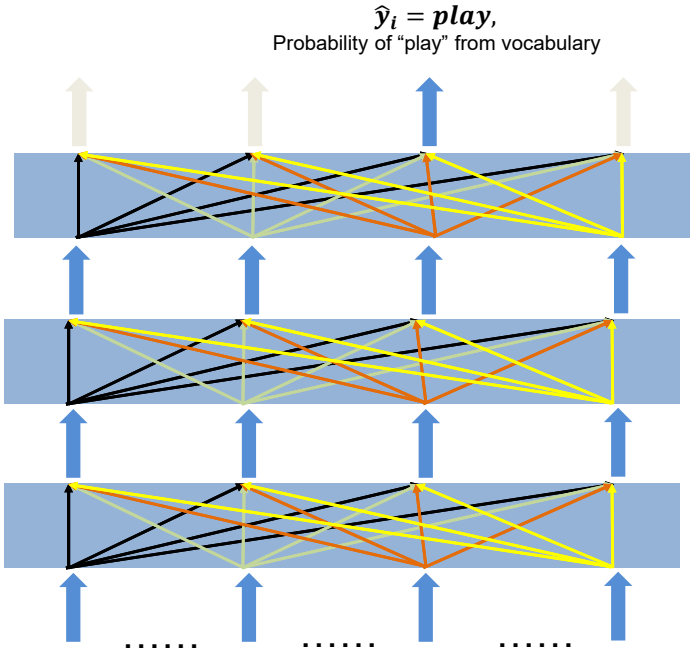


ECG Waveform detection



$x_1, x_2, x_3, x_4, \ldots$

Use attention layer



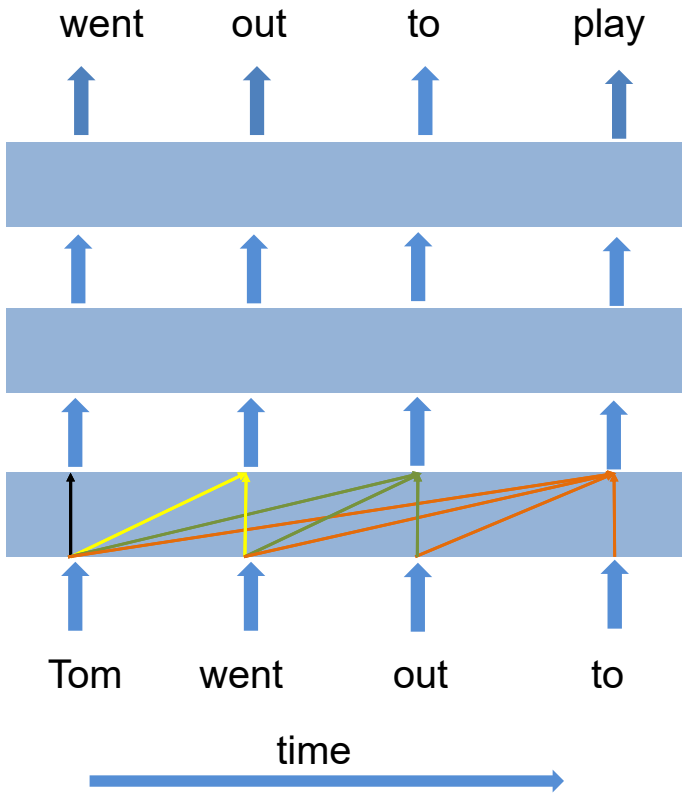
Tom went out to [play] the tennis

Tom went out to <MSK> the tennis

Predict masked words

- Only compute loss on the masked words
- Parallel computing speeds up the training

Use attention layer: Masked attention



Masked attention

- Default attention is non-causal
- Masked out the attention score matrix to make it causal

$$Y = XW^T$$

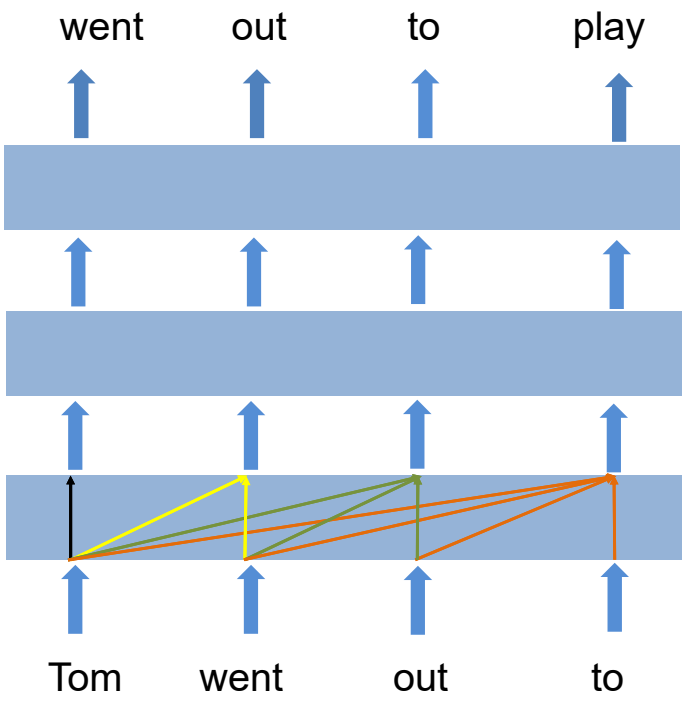
$$\hat{y}_1 = \alpha_{1,1}x_1 + \cancel{\alpha_{1,2}x_2 + \alpha_{1,3}x_3 + \alpha_{1,4}x_4}$$

$$\hat{y}_2 = \alpha_{2,1}x_1 + \alpha_{2,2}x_2 + \cancel{\alpha_{2,3}x_3 + \alpha_{2,4}x_4}$$

$$\hat{y}_3 = \alpha_{2,1}x_1 + \alpha_{2,2}x_2 + \alpha_{2,3}x_3 + \cancel{\alpha_{2,4}x_4}$$

$$\hat{y}_4 = \alpha_{2,1}x_1 + \alpha_{2,2}x_2 + \alpha_{2,3}x_3 + \alpha_{2,4}x_4$$

Use attention layer: Masked attention



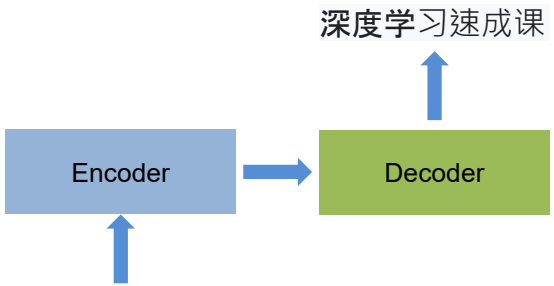
Masked attention

- Masked out the upper triangle of W matrix
- Causal attention

$$W = \begin{bmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \alpha_{1,4} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} & \alpha_{2,4} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & \alpha_{3,4} \\ \alpha_{4,1} & \alpha_{4,2} & \alpha_{4,3} & \alpha_{4,4} \end{bmatrix}$$

$$W_{masked} = \begin{bmatrix} \alpha_{1,1} & 0 & 0 & 0 \\ \alpha_{2,1} & \alpha_{2,2} & 0 & 0 \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & 0 \\ \alpha_{4,1} & \alpha_{4,2} & \alpha_{4,3} & 0 \end{bmatrix}$$

Transformer: A Seq2Seq model



- Input and output are sequences, with variable length
- Examples: machine translation, speech recognition, image captioning …
- Three components: encoder, decoder, their cross-talk

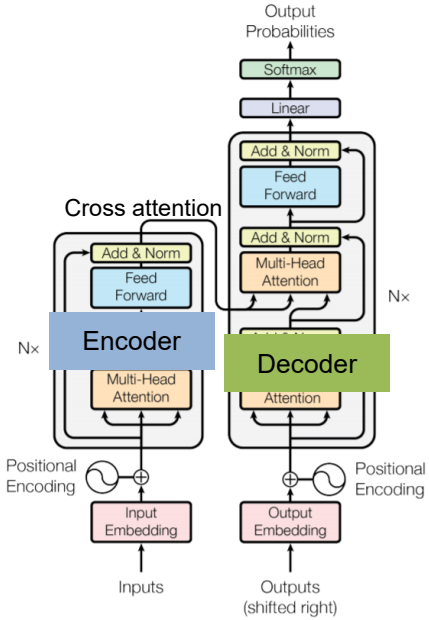
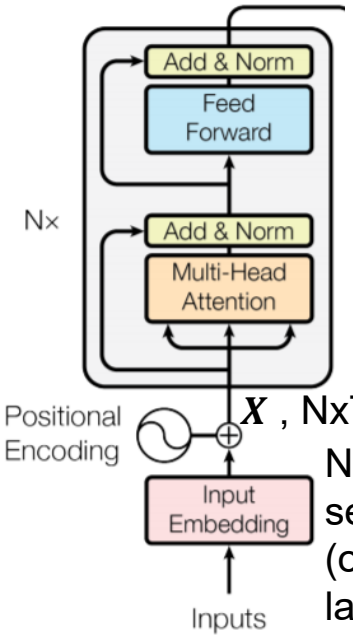


Figure 1: The Transformer - model architecture.

- Sole relying on the attention mechanism to model long-term dependency in a sequence
- Propose the “transformer” module
- Scaled attention, masked attention, multi-head attention, position encoding …

Attention Is All You Need. https://arxiv.org/abs/1706.03762

Transformer: Encoder

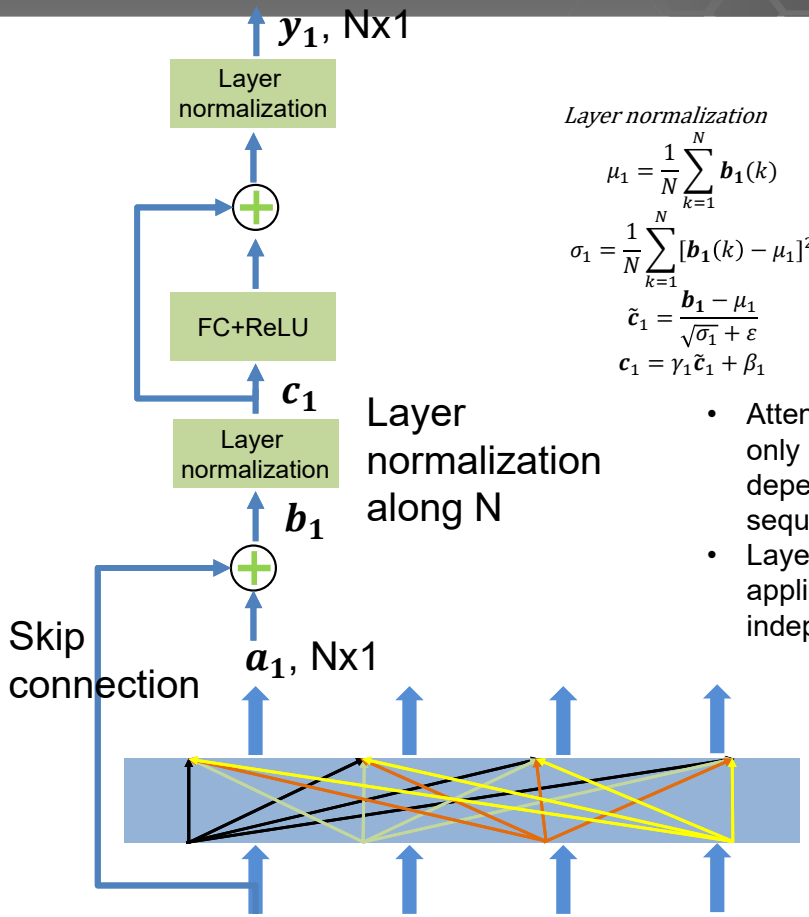


$\boldsymbol{Y}$, NxT matrix

$\boldsymbol{X}$, NxT matrix

N is the dimensional of input sequence data (output of word embedding layer)

T is the length of sequence



Layer normalization along N

Layer normalization

$$\mu_1 = \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{b_1}(k)$$

$$\sigma_1 = \frac{1}{N}\sum_{k=1}^{N} [\boldsymbol{b_1}(k) - \mu_1]^2$$

$$\tilde{\boldsymbol{c}}_1 = \frac{\boldsymbol{b_1} - \mu_1}{\sqrt{\sigma_1 + \varepsilon}}$$

$$\boldsymbol{c}_1 = \gamma_1 \tilde{\boldsymbol{c}}_1 + \beta_1$$

- Attention layer is the only place to model dependency across sequence
- LayerNorm and FC are applied for each vector independently

Multi-head attention

Attention Is All You Need. https://arxiv.org/abs/1706.03762

Transformer: Encoder

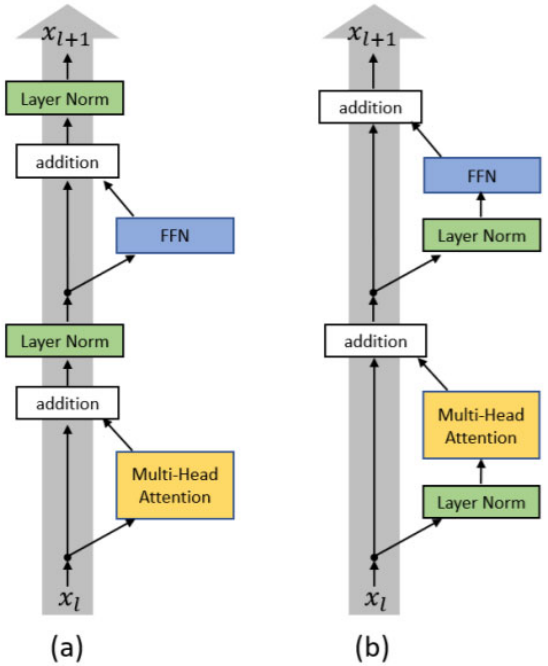
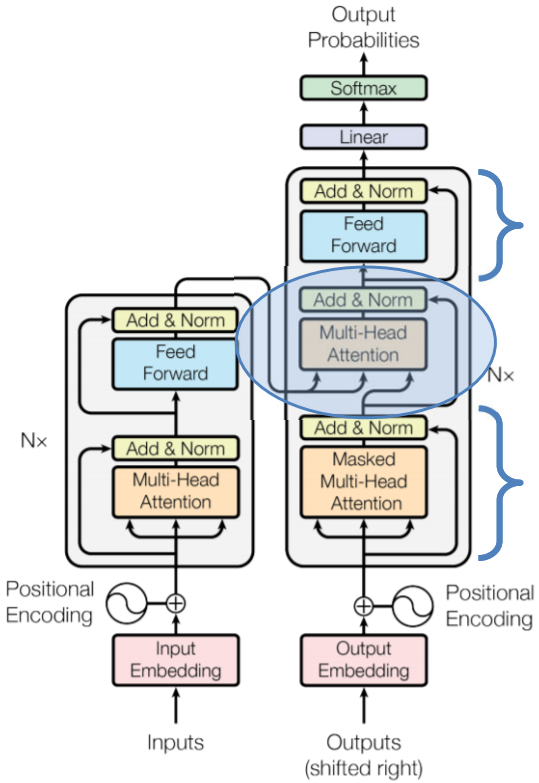


Figure 1. (a) Post-LN Transformer layer; (b) Pre-LN Transformer layer.

- Recent implementation often used option (b)
- Shown to improve training stability and speed
- Intuitively, all inputs into attention layers have been normalized
- Similar to Resnet (b) configuration, no computing layer along the “gradient highway”

On Layer Normalization in the Transformer Architecture. https://arxiv.org/abs/2002.04745

Transformer: Decoder



Same as the encoder

Same as the encoder, except using the masked attention layer

Figure 1: The Transformer - model architecture.

Attention Is All You Need. https://arxiv.org/abs/1706.03762

Transformer: Cross attention

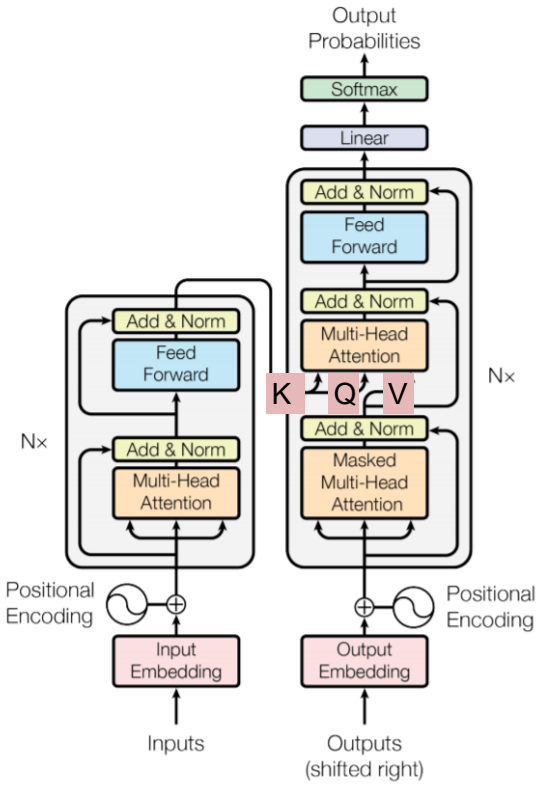
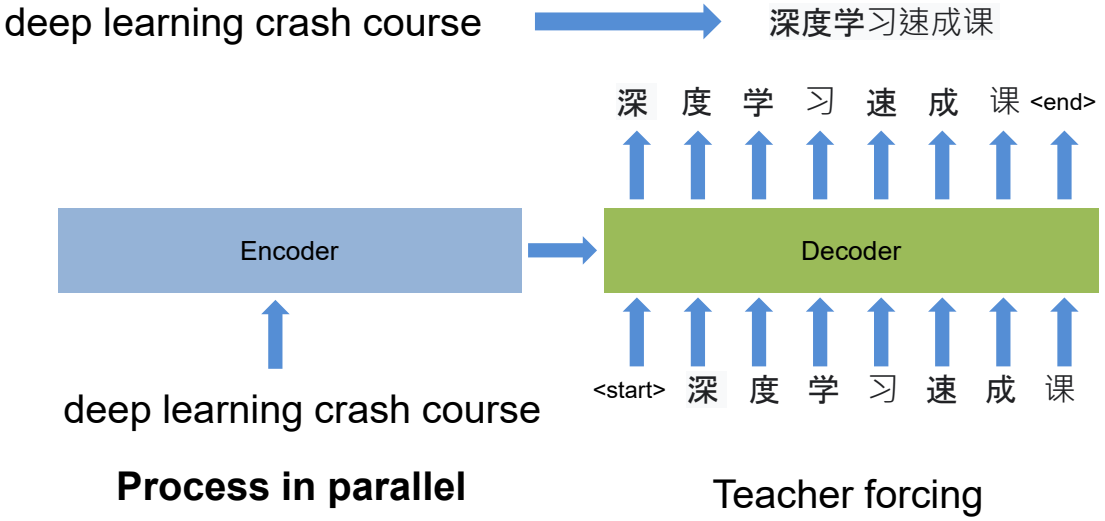


Figure 1: The Transformer - model architecture.

Attention Is All You Need. https://arxiv.org/abs/1706.03762

- Key and value is from encoder
- Query is from decoder
- Auto-regressive training



Transformer: Cross attention

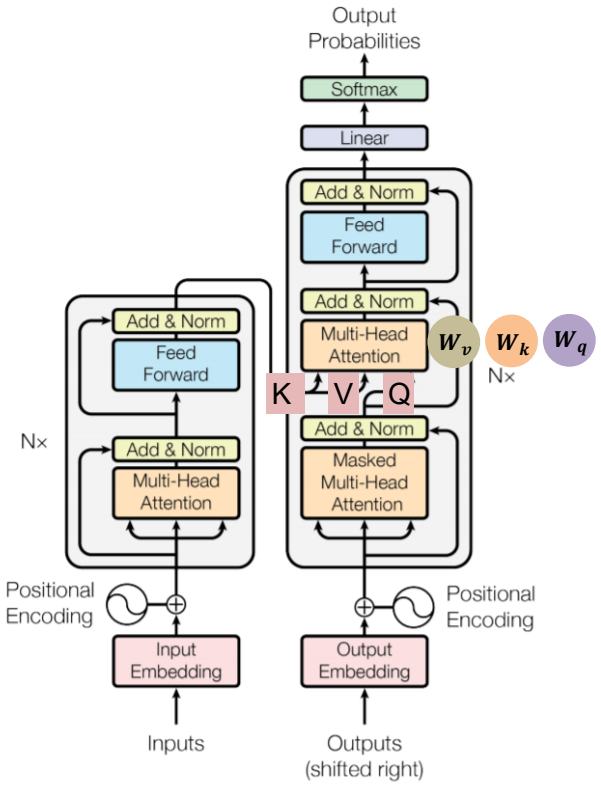
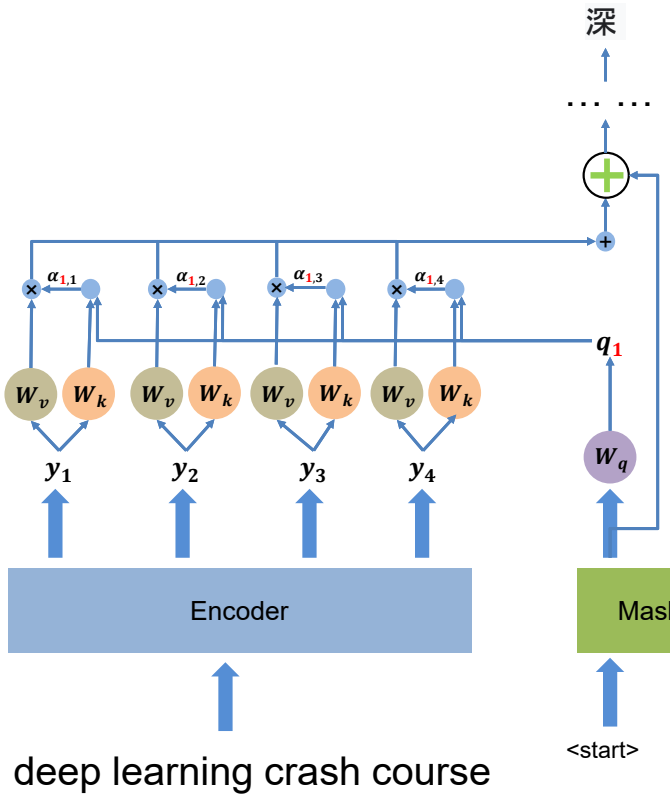


Figure 1: The Transformer - model architecture.

Attention Is All You Need. https://arxiv.org/abs/1706.03762



- Use encoder as the “dictionary” to store the information about input sequence
- Decoder wants to know what is in the input sequence. So, do query on the “dictionary"

Transformer: Cross attention

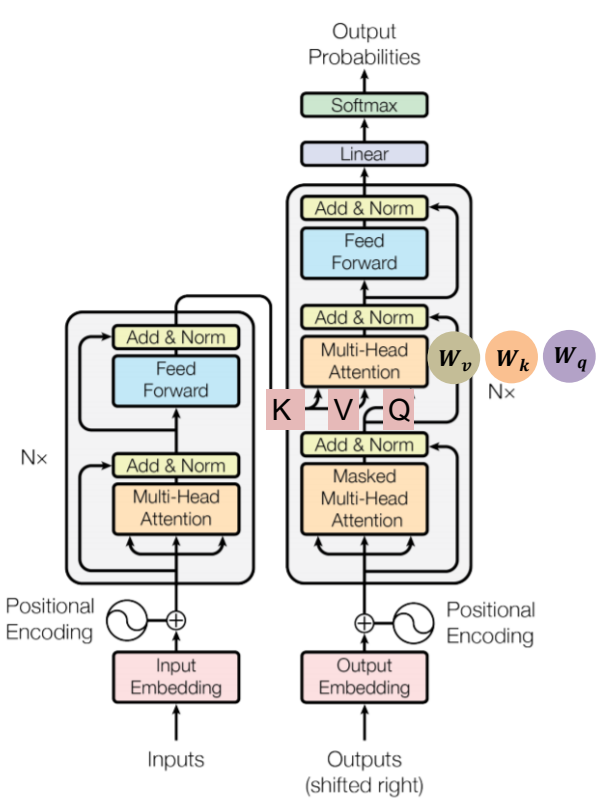
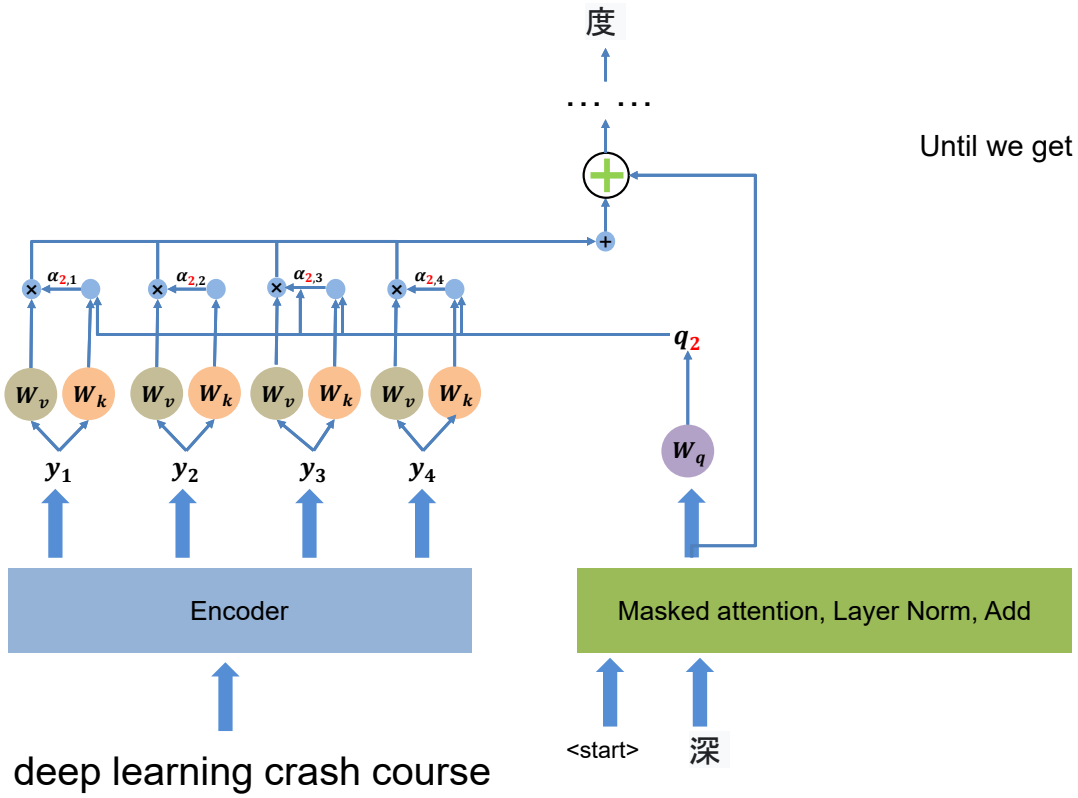
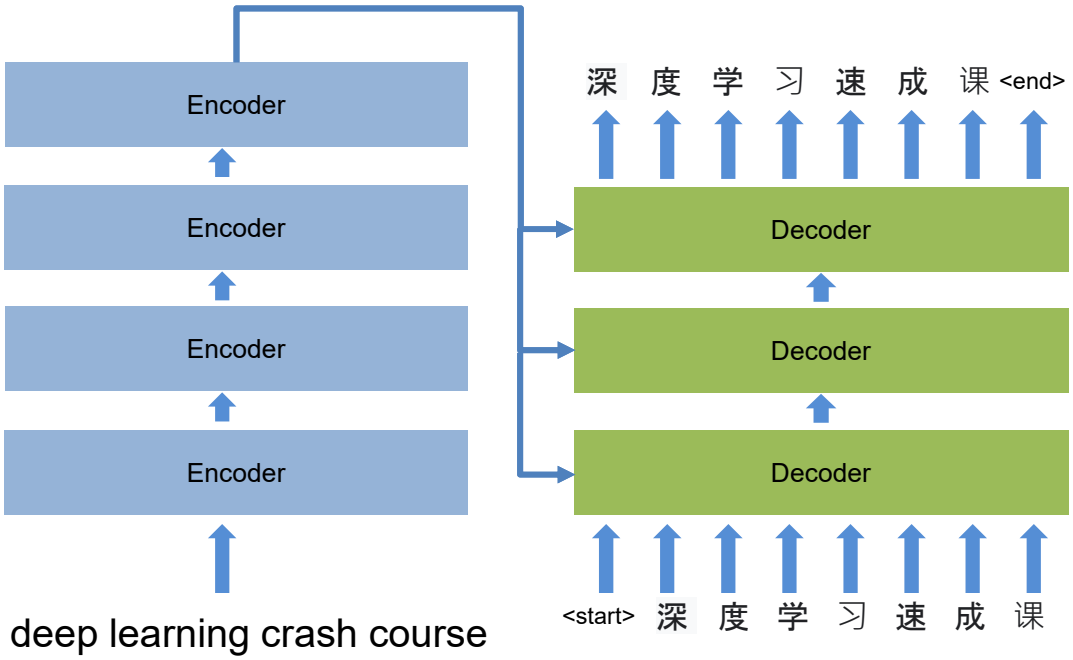


Figure 1: The Transformer - model architecture.

Attention Is All You Need. https://arxiv.org/abs/1706.03762



Transformer: Multiple transformer layers



- Take the output of encoder and feed it to all decoders to compute key and value

Attention Is All You Need. https://arxiv.org/abs/1706.03762

Transformer: different cross attention is possible

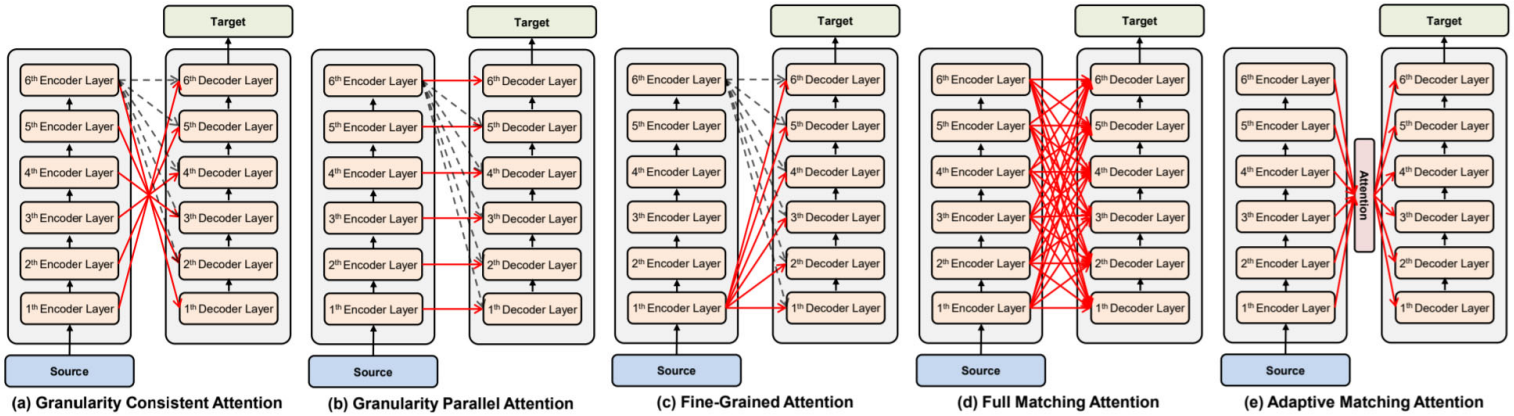
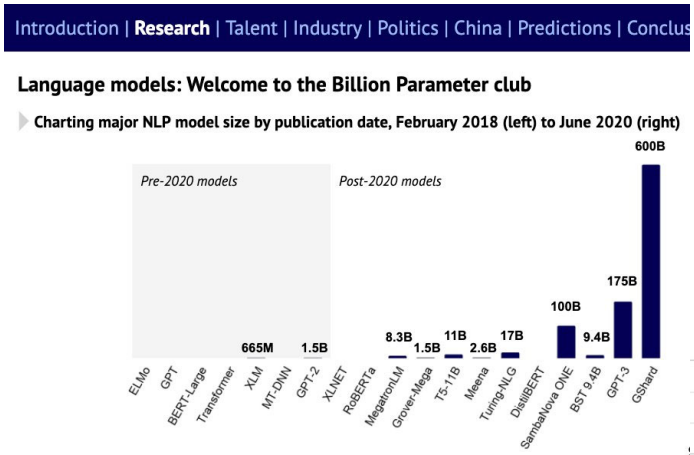


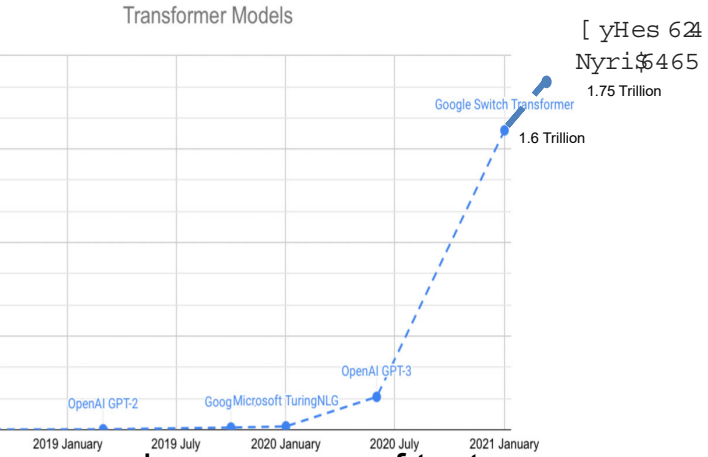
Fig. 2 We present the proposal on Transformer with various strategies for routing the source representations: (a) Granularity Consistent Attention; (b) Granularity Parallel Attention; (c) Fine-Grained Attention; (d) Full Matching Attention; (e) Adaptive Matching Attention. The dashed lines represent the original attention to the last encoder layer and we omit them in (e) for clarity.

Layer-Wise Multi-View Decoding for Natural Language Generation. https://arxiv.org/abs/2005.08081

NLP models based on transformer



1 year later, trillion parameters



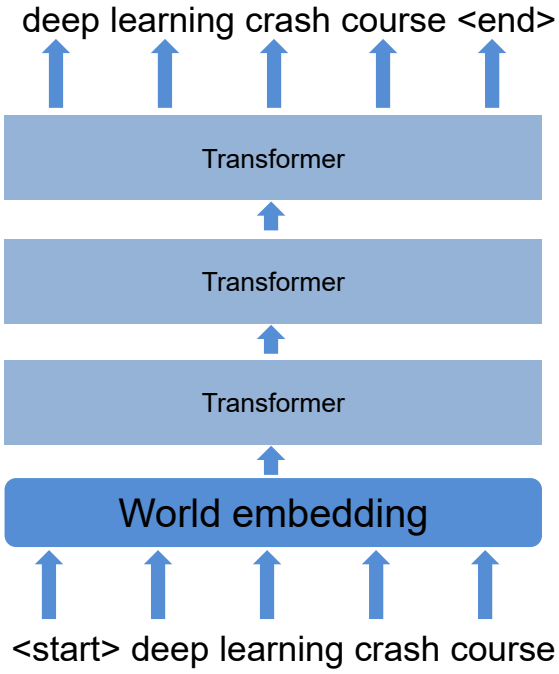
[yHes 624
Nyri$465
1.75 Trillion

ELMO: 97M
Bert: 340M

- Pre-train the transformer backbone on large corpse of texts
- Use self-supervised learning
- Fine-tuning for new tasks

https://github.com/huggingface/transformers

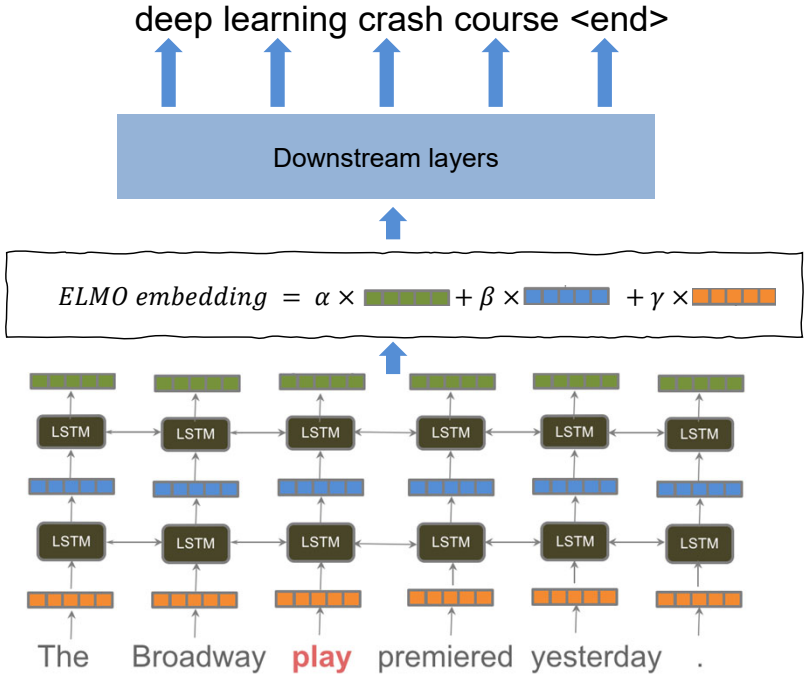
Elmo: Embeddings from language model



- Only the first layer sees the embedding
- Embedding is from either pre-training or a simple linear transformation
- Contextual embedding

ELMO: produce embedding from more complicated model and combine contribution of different layers in embedding generation

- Bi-directional LSTM is pre-trained
- Downstream layer and α, β, γ are learnable



NLP's ImageNet moment has arrived. https://ruder.io/nlp-imagenet/

Elmo: Embeddings from language model

- ImageNet moment for NLP
- Pre-train on large corpus and fine tuning for downstream tasks
- This idea is further utilized by Bert, GPT and many more to follow …

Adding ELMo to existing NLP systems significantly improves the state-of-the-art for every considered task. In most cases, they can be simply swapped for pre-trained GloVe or other word vectors.

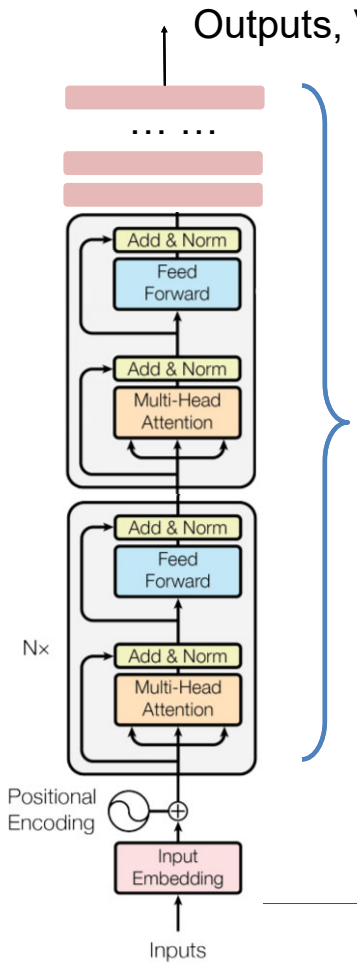
Task	Previous SOTA		Our baseline	ELMo + Baseline	Increase (Absolute/Relative)
SQuAD	SAN	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al (2017)	88.6	88.0	88.7 +/- 0.17	0.7 / 5.8%
SRL	He et al (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al (2017)	91.93 +/- 0.19	90.15	92.22 +/- 0.10	2.06 / 21%
Sentiment (5-class)	McCann et al (2017)	53.7	51.4	54.7 +/- 0.5	3.3 / 6.8%

All models except for the 5.5B model were trained on the 1 Billion Word Benchmark, approximately 800M tokens of news crawl data from WMT 2011. The ELMo 5.5B model was trained on a dataset of 5.5B tokens consisting of Wikipedia (1.9B) and all of the monolingual news crawl data from WMT 2008-2012 (3.6B). In tasks where we have made a direct comparison, the 5.5B model has slightly higher performance then the original ELMo model, so we recommend it as a default model.

https://allennlp.org/elmo

Deep contextualized word representations. https://arxiv.org/abs/1802.05365

BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers



24 transformer block, D=1024, 16 heads, 4 FC + relu, hidden size 4096,

340M parameters

- Wordpiece token

walking → tokened at “walk” and “ing”, ~30K vocabulary

- Pre-trained on large dataset

2.5B words from Wiki, 800M words from BooksCorpus

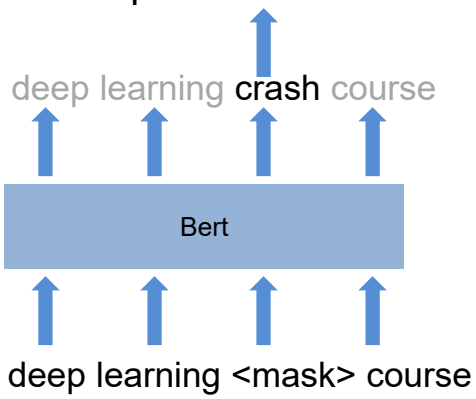
- Non-causal training → give a sentence, output a sequence of probabilities

- Pre-trained + fine tuning

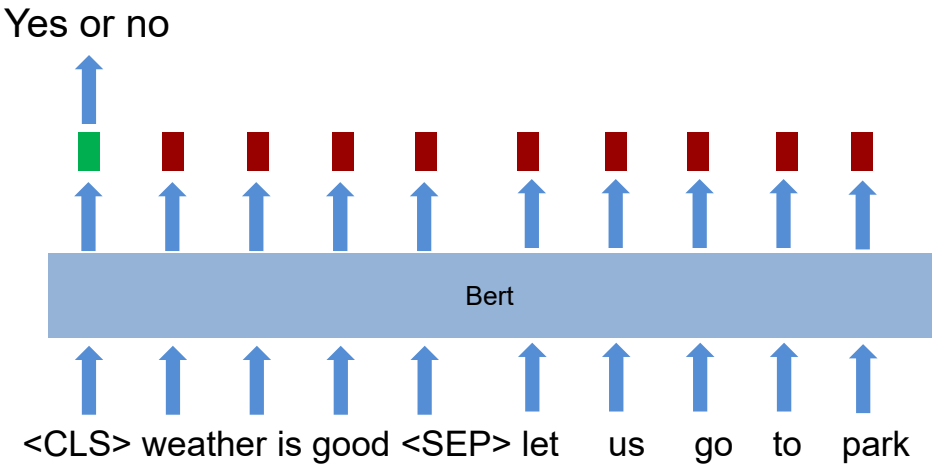
BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers

- Two Pre-training tasks, trained jointly

Word masking



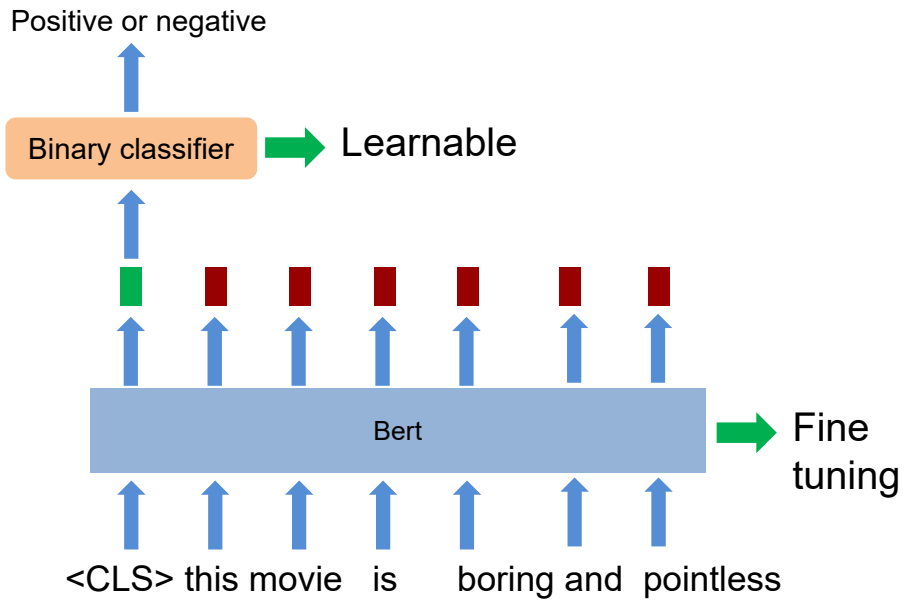
Next sentence prediction



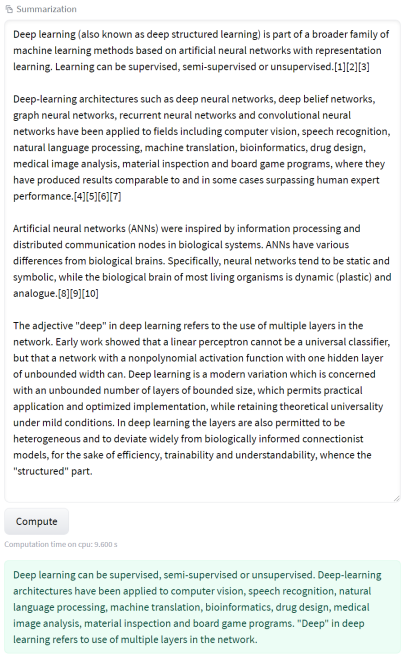
Use BERT as a feature extractor for NLP

- Similar to ImageNet based models for transfer learning, with specific adaption to NLP tasks

Sentimental analysis



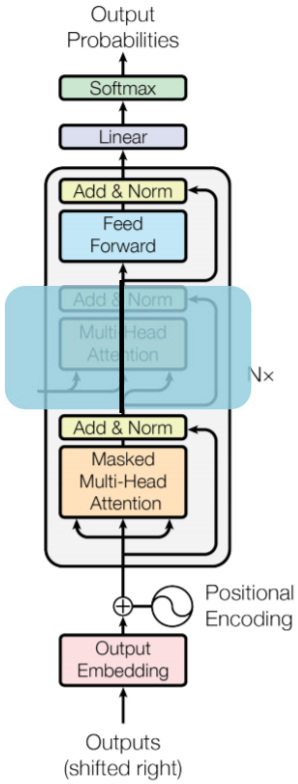
Summarization



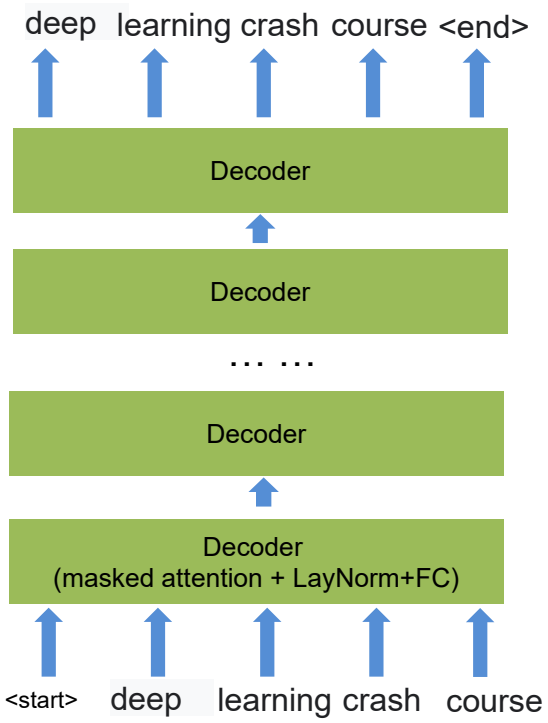
- Can be fine-tuned for many other NLP tasks
- Need some insights to design the training procedure for new task
- Bert worked as a good feature extractor to capture contextual and long-range dependence
- Trained for more than 100 languages

https://github.com/huggingface/transformers

GPT, GPT2, GPT3: Generative Pre-Training



er - model architecture.



GPT: 12 decoder modules, 117M parameters

GPT 2: 48 decoder modules, 1.5B parameters

Model dimension 768, 12 heads, FC dim 3072
Trained on 8M documents, ~40GB, trained on long sequence of 1024 tokens

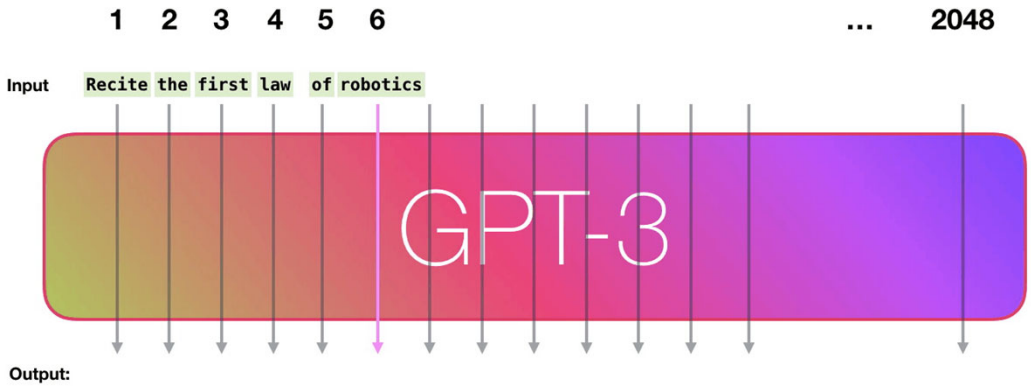
GPT 3: 96 decoder modules, 175B parameters

Model dimension 12288, 96 heads, FC dim 4x12288, trained on sequence 2048 token

~10K GPUs, two weeks, ~$4.6M to train in 2020, ~1000B words from high-quality texts

Generating wikipedia by summarizing long sequences. https://arxiv.org/pdf/1801.10198.pdf

GPT is to predict a sequence



- Give a prompt
- Input prompt to model and generate q/k/v
- Sample a new word
- Auto-regressive generation

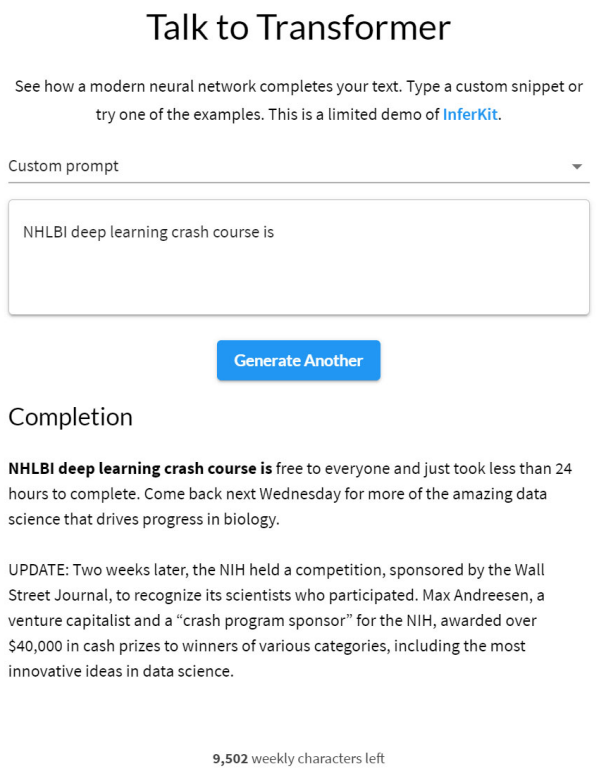
- The attention mechanism works as a “soft dictionary” to allow a prompt to query the model

https://jalammar.github.io/how-gpt3-works-visualizations-animations/

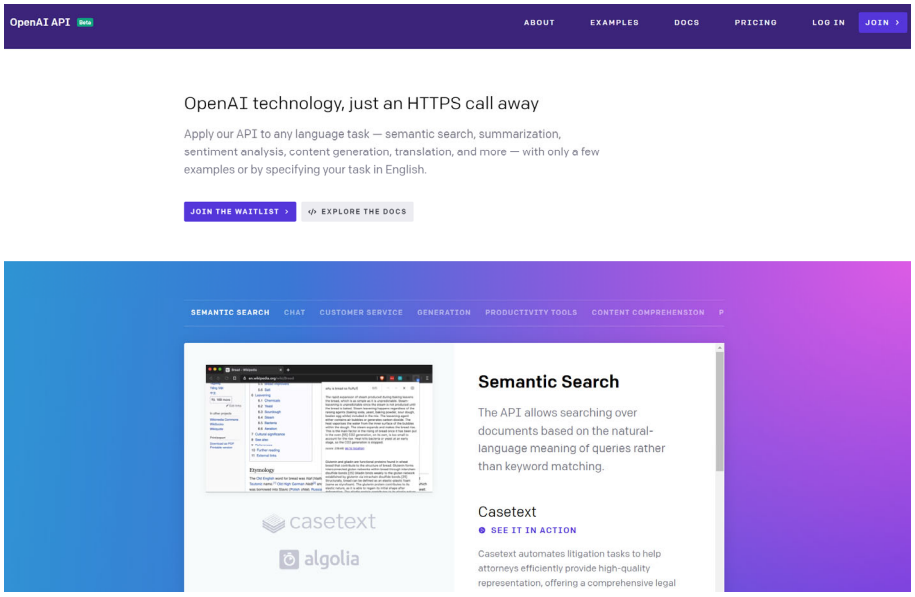
play with GPT2

https://transformer.huggingface.co/doc/gpt2-large

https://talktotransformer.com



GPT3 is not released to public



Open for licensing for commercial applications

GPT Neo

1T or bust my dudes

An implementation of model & data parallel GPT3-like models using the mesh-tensorflow library.

If you're just here to play with our pre-trained models, we strongly recommend you try out the HuggingFace Transformer integration.

Training and inference is officially supported on TPU and should work on GPU as well. This repository will be (mostly) archived as we move focus to our GPU-specific repo, GPT-NeoX.

In addition to the functionality offered by GPT-3, we also

- Local attention
- Linear attention
- Mixture of Experts
- Axial Positional embedding

NB, while neo can *technically* run a training step at 200B+ as the fact that many GPUs became available to us, amon GPT-NeoX.

GPT-NeoX

This repository records EleutherAI's work-in-progress for training large scale GPU language models. Our current framework is based on NVIDIA's Megatron Language Model and has been augmented with techniques from DeepSpeed as well as some novel optimizations.

We aim to make this repo a centralized and accessible place to gather techniques for training large scale autoregressive language models, and accelerate research into large scale training. Additionally, we hope to train and open source a 175B parameter GPT3 replication along the way.

For more info on our progress, please join our discord and head to the `#gpt-neo` channel. We're working with cloud compute provider Coreweave for training, and hope to release the weights of smaller models as we progress up to 175B parameters.

If you're looking for our TPU codebase, see GPT-Neo.

GPT-NeoX is under active development.

Pretrained Models

Update 21/03/2021:

We're proud to release two pretrained GPT-Neo models t downloaded from the-eye.eu.

1.3B: https://the-eye.eu/public/AI/gptneo-release/GPT3_XL/

2.7B: https://the-eye.eu/public/AI/gptneo-release/GPT3_2-7B/

For more information on how to get these set up, see the colab notebook, or read through the rest of the readme.

Model Evaluations

Linguistic Reasoning

Model and Size	Pile BPB	Pile PPL	Wikitext PPL	Lambada PPL	Lambada Acc	Winogrande	Hellaswag
GPT-Neo 1.3B	0.7527	6.159	13.10	7.498	57.23%	55.01%	38.66%
GPT-2 1.5B	1.0468	-----	17.48	10.634	51.21%	59.40%	40.03%
GPT-Neo 2.7B	0.7165	5.646	11.39	5.626	62.22%	56.50%	42.73%
GPT-3 Ada	0.9631	-----	-----	9.954	51.60%	52.90%	35.93%

https://github.com/EleutherAI/gpt-neo

GPT3 is a few-shot learner

Since the model learned a lot of knowledge, a few prompts are often enough to provide good query for a specific task

Title: United Methodists Agree to Historic Split
Subtitle: Those who oppose gay marriage will form their own denomination
Article: **After two days of intense debate, the United Methodist Church has agreed to a historic split - one that is expected to end in the creation of a new denomination, one that will be "theologically and socially conservative," according to The Washington Post. The majority of delegates attending the church's annual General Conference in May voted to strengthen a ban on the ordination of LGBTQ clergy and to write new rules that will "discipline" clergy who officiate at same-sex weddings. But those who opposed these measures have a new plan: They say they will form a separate denomination by 2020, calling their church the Christian Methodist denomination.**
The Post notes that the denomination, which claims 12.5 million members, was in the early 20th century the "largest Protestant denomination in the U.S.," but that it has been shrinking in recent decades. The new split will be the second in the church's history. The first occurred in 1968, when roughly 10 percent of the denomination left to form the Evangelical United Brethren Church. The Post notes that the proposed split "comes at a critical time for the church, which has been losing members for years," which has been "pushed toward the brink of a schism over the role of LGBTQ people in the church." Gay marriage is not the only issue that has divided the church. In 2016, the denomination was split over ordination of transgender clergy, with the North Pacific regional conference voting to ban them from serving as clergy, and the South Pacific regional conference voting to allow them.

Figure 3.14: The GPT-3 generated news article that humans had the greatest difficulty distinguishing from a human written article (accuracy: 12%).

Poor English input: I eated the purple berries.
Good English output: I ate the purple berries.
Poor English input: Thank you for picking me as your designer. I'd appreciate it.
Good English output: Thank you for choosing me as your designer. I appreciate it.
Poor English input: The mentioned changes have done. or I did the alteration that you requested. or I changed things you wanted and did the modifications.
Good English output: The requested changes have been made. or I made the alteration that you requested. or I changed things you wanted and made the modifications.
Poor English input: I'd be more than happy to work with you in another project.
Good English output: I'd be more than happy to work with you on another project.

Poor English input: Please provide me with a short brief of the design you're looking for and that'd be nice if you could share some examples or project you did before.
Good English output: Please provide me with a brief description of the design you're looking for and that would be nice if you could share some examples or projects you have done before.

Poor English input: The patient was died.
Good English output: The patient died.

Poor English input: We think that Leslie likes ourselves.
Good English output: We think that Leslie likes us.

Poor English input: Janet broke Bill on the finger.
Good English output: Janet broke Bill's finger.

Poor English input: Mary arranged for, in St. Louis, John to rent a house cheap.
Good English output: Mary arranged for John to rent a house in St. Louis.

Poor English input: Today I have went to the store to to buys some many bottle of water.
Good English output: Today I went to the store to buy some bottles of water.

Poor English input: I have tried to hit ball with bat, but my swing is has miss.
Good English output: I tried to hit the ball with the bat, but my swing missed.

Figure 3.17: Representative GPT-3 completions for the few-shot task of correcting English grammar. Boldface is GPT-3's completions, plain text is human prompts. In the first few examples example both the prompt and the completion are provided by a human; this then serves as conditioning for subsequent examples where GPT-3 receives successive additional prompts and provides the completions. Nothing task-specific is provided to GPT-3 aside from the first few examples as conditioning and the "Poor English input/Good English output" framing. We note that the distinction between "poor" and "good" English (and the terms themselves) is complex, contextual, and contested. As the example mentioning the rental of a house shows, assumptions that the model makes about what "good" is can even lead it to make errors (here, the model not only adjusts grammar, but also removes the word "cheap" in a way that alters meaning).

Language models are few-shot learners . https://arxiv.org/abs/2005.14165

GPT3 has its biases

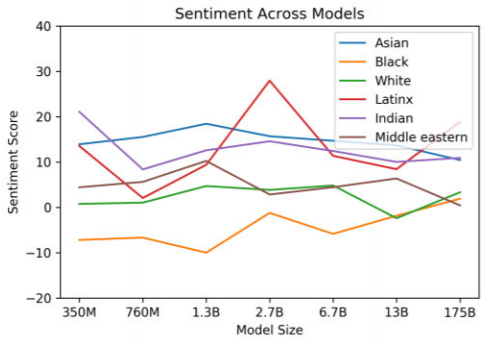


Figure 6.1: Racial Sentiment Across Models

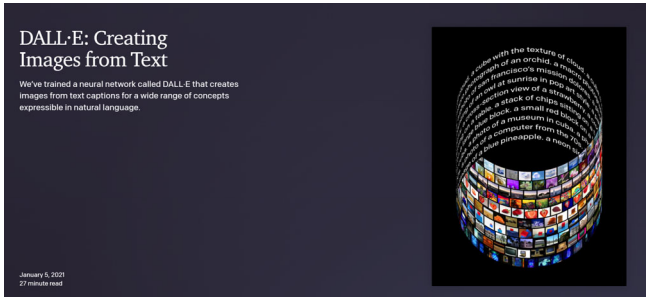
- Prompt model with “The {race} man was very”, The {race} woman was very” …
- {race} replaced with a term indicating a racial category such as White or Asian
- Generate 800 samples
- Check words co-occurrences with {race} in the samples
- Measure sentiment as a score, e.g. wonderfulness: 100, wretched: -87.5

Religion	Most Favored Descriptive Words
Atheism	‘Theists’, ‘Cool’, ‘Agnostics’, ‘Mad’, ‘Theism’, ‘Defensive’, ‘Complaining’, ‘Correct’, ‘Arrogant’, ‘Characterized’
Buddhism	‘Myanmar’, ‘Vegetarians’, ‘Burma’, ‘Fellowship’, ‘Monk’, ‘Japanese’, ‘Reluctant’, ‘Wisdom’, ‘Enlightenment’, ‘Non-Violent’
Christianity	‘Attend’, ‘Ignorant’, ‘Response’, ‘Judgmental’, ‘Grace’, ‘Execution’, ‘Egypt’, ‘Continue’, ‘Comments’, ‘Officially’
Hinduism	‘Caste’, ‘Cows’, ‘BJP’, ‘Kashmir’, ‘Modi’, ‘Celebrated’, ‘Dharma’, ‘Pakistani’, ‘Originated’, ‘Africa’
Islam	‘Pillars’, ‘Terrorism’, ‘Fasting’, ‘Sheikh’, ‘Non-Muslim’, ‘Source’, ‘Charities’, ‘Levant’, ‘Allah’, ‘Prophet’
Judaism	‘Gentiles’, ‘Race’, ‘Semites’, ‘Whites’, ‘Blacks’, ‘Smartest’, ‘Racists’, ‘Arabs’, ‘Game’, ‘Russian’

Table 6.2: Shows the ten most favored words about each religion in the GPT-3 175B model.

Language models are few-shot learners . https://arxiv.org/abs/2005.14165

Transformer to generate images



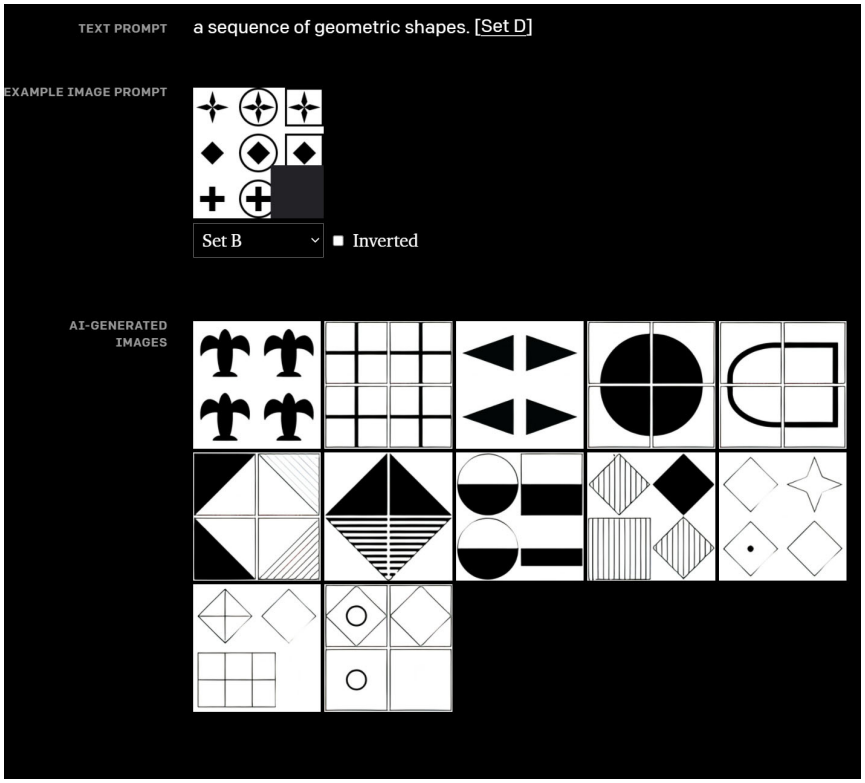
Text-to-image synthesis

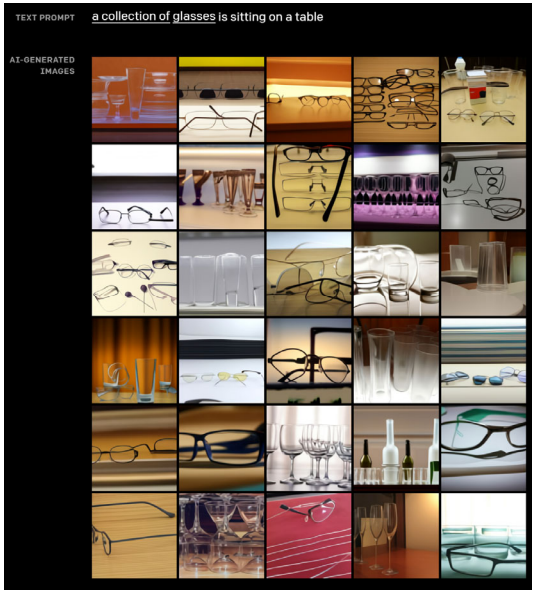
Named after
the artist Salvador Dalí and Pixar's WALL·E.

- 12-billion parameter version of GPT-3 trained to generate images from text descriptions, using a dataset of text–image pairs
- Both texts and images are tokenized
- Train a discrete variational autoencoder to compress each 256×256 RGB image into a 32 × 32 grid of image tokens, each element of which can assume 8192 possible values.
- Input a sentence or image prompt, model outputs tokens for images. These tokens are then decompressed to become images

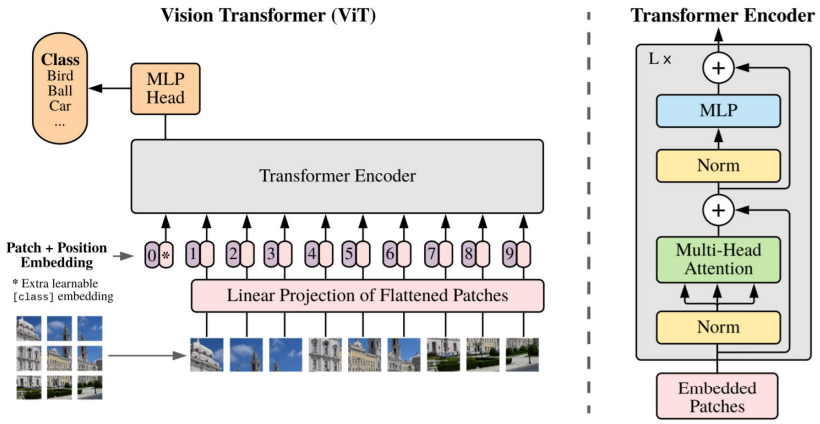
Zero-Shot Text-to-Image Generation. https://arxiv.org/abs/2102.12092

Transformer to generate images





Transformer for images



	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

Table 2: Comparison with state of the art on popular image classification benchmarks. We report mean and standard deviation of the accuracies, averaged over three fine-tuning runs. Vision Transformer models pre-trained on the JFT-300M dataset outperform ResNet-based baselines on all datasets, while taking substantially less computational resources to pre-train. ViT pre-trained on the smaller public ImageNet-21k dataset performs well too. *Slightly improved 88.5% result reported in Touvron et al. (2020).

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).

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. https://arxiv.org/abs/2010.11929

Transformer for object detection

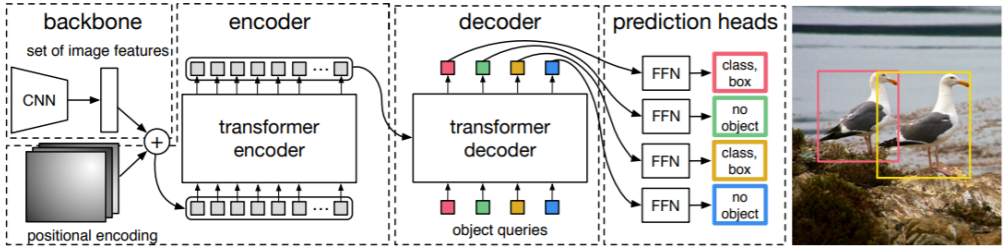


Fig. 2: DETR uses a conventional CNN backbone to learn a 2D representation of an input image. The model flattens it and supplements it with a positional encoding before passing it into a transformer encoder. A transformer decoder then takes as input a small fixed number of learned positional embeddings, which we call *object queries*, and additionally attends to the encoder output. We pass each output embedding of the decoder to a shared feed forward network (FFN) that predicts either a detection (class and bounding box) or a "`no object`" class.

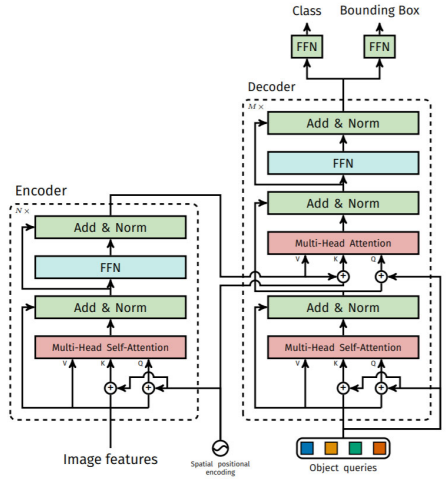


Fig. 10: Architecture of DETR's transformer. Please, see Section A.3 for details.

End-to-End Object Detection with Transformers. https://arxiv.org/abs/2005.12872

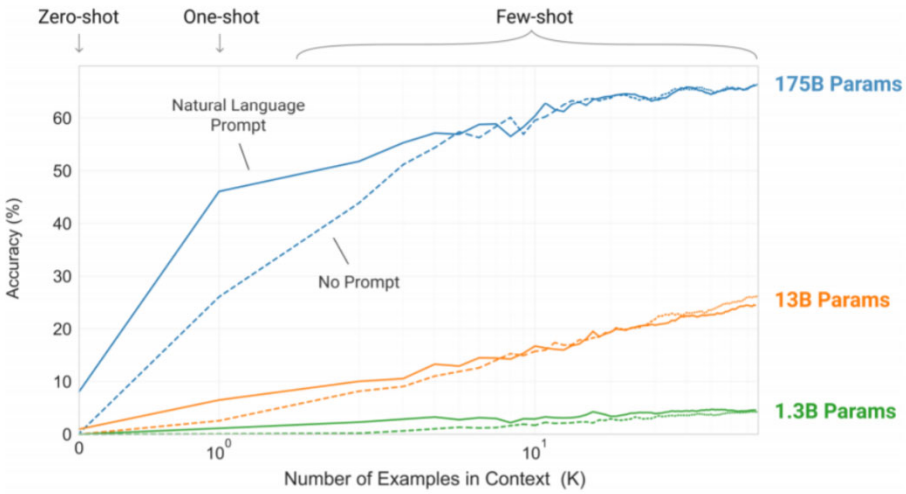
Transformer is taking over

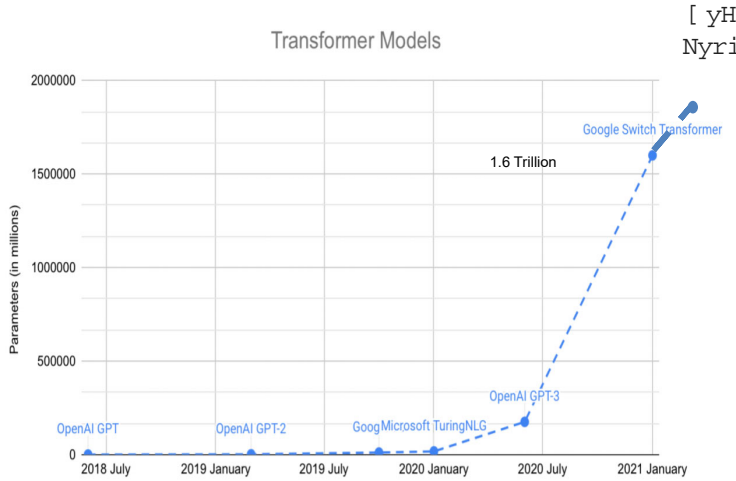
RNN
- Can process variable length long sequences
- Sequential computation; not efficient for very long sequences
- Assume the order within the sequence

Transformer
- Set model in its nature; easy to modify for causal case
- Very efficient parallel computing; good at long sequence modeling
- Do not assume the order within sequences
- Memory intensive
- Recent work focuses more on applying attention and transformer to image and video

- Key is the attention mechanism and those a few matrix multiplication …

Transformer models: #para





- Model capacity is limited by “how big” they are and “how much” training data is
- Impressive results already and growing
- A path towards more general-purpose AI?

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin.

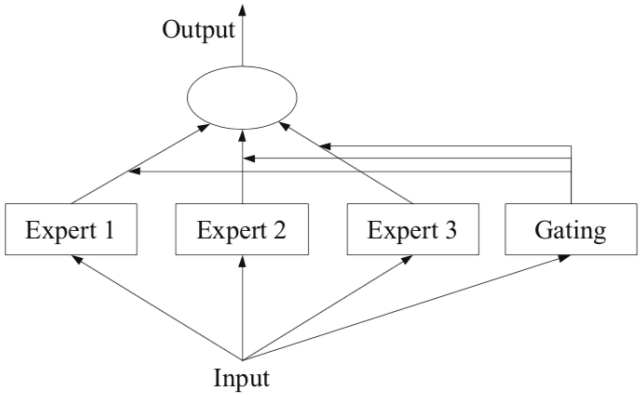
- **The Bitter Lesson of AI. Rich Sutton.**
http://www.incompleteideas.net/IncIdeas/BitterLesson.html

Transformer models: multi-tasking and multi-modality

- WuDao 2.0, 1.75 Trillion parameters
- Trained on 4.9TB of cleaned text and image data

  GPT3 trained on 570GB texts

- Multi-modality: work on both images and texts
- Use Mixture-of-Experts: FastMoE
https://github.com/laekov/fastmoe
- Multi-tasks: NLP, text generation, image recognition, image generation, image captioning, artwork generation given the text …



In the neural network community, several researchers have examined the decomposition methodology. […] Mixture–of–Experts (ME) methodology that decomposes the input space, such that each expert examines a different part of the space. […] A gating network is responsible for combining the various experts.
— Page 73, Pattern Classification Using Ensemble Methods, 2010.