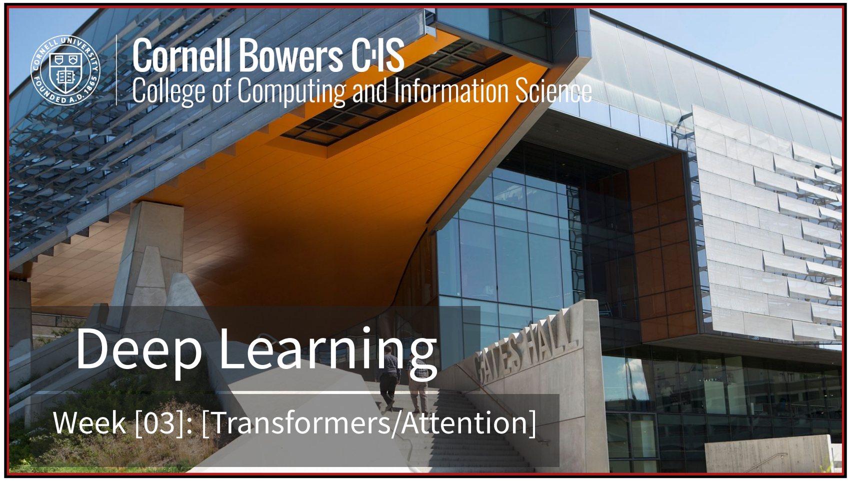
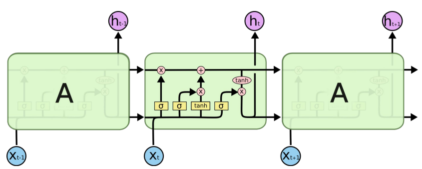


Logistics

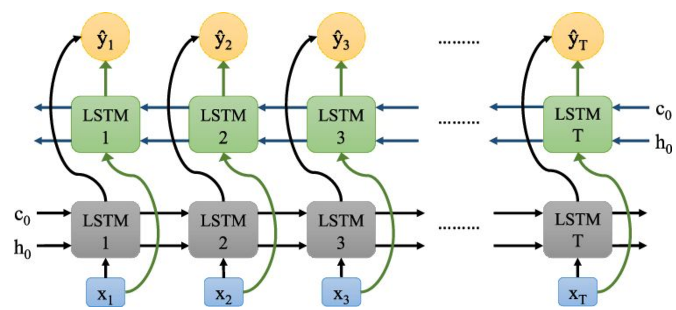
- HW2 is out
- We have a feedback form (due Friday February 23)
- Tuesday 10am office hours might change
- We will talking about projects on Thursday
  - HW3 will be a shorter

Previously: Using LSTMs to solve sequence problems

- Process sequences one element at a time.
- Maintain a 'memory' (cell state) to capture information about previous steps.
- Mitigates the RNN vanishing gradient problem
- Suitable for time series, speech, text, and other sequential data.

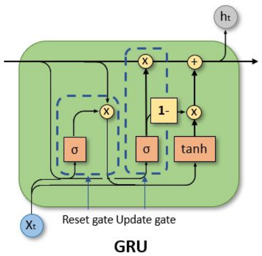


Bidirectional LSTM



Cornell Bowers CIS

Gated recurrent units (GRUs)

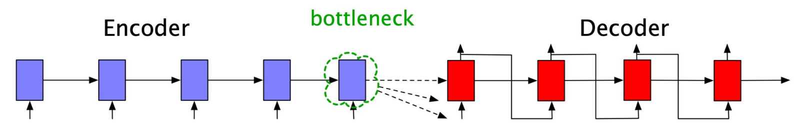


$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$
$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$
$$\hat{h}_t = \phi(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$
$$h_t = (1 - z_t) \odot \hat{h}_t + z_t \odot h_{t-1}$$

Cornell Bowers CIS

Bottleneck Problem

- All the information about the source sequence must be stored in a single vector
  - How to translate a long paragraph?
  - How to summarize long articles?

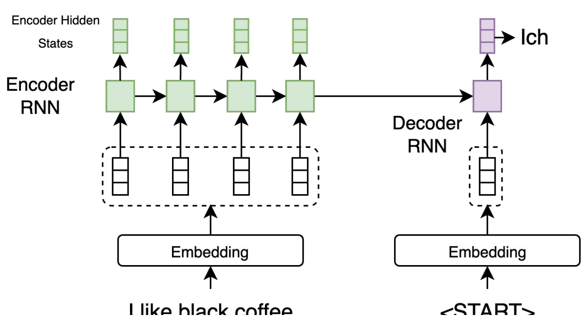


https://web.stanford.edu/~jurafsky/slp3/

Cornell Bowers CIS

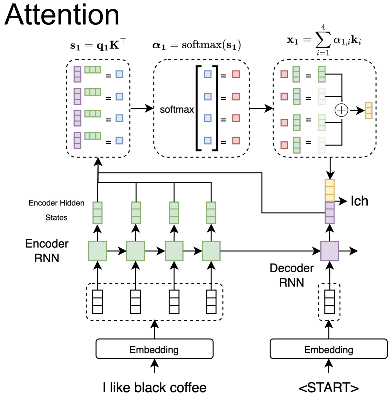
RNN for Machine Translation

Would be nice if we could "look back" at previous hidden states

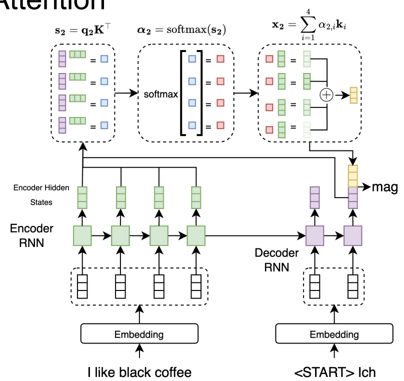


Cornell Bowers CIS

RNN with Attention

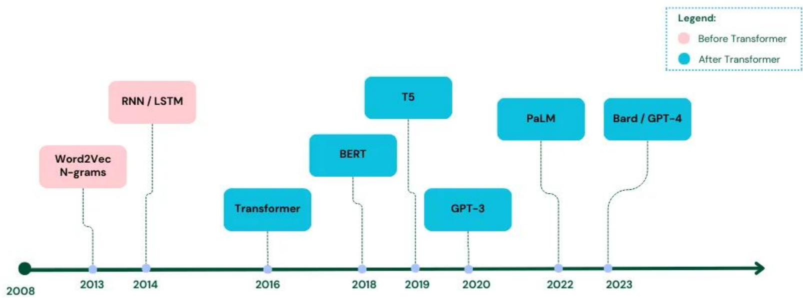


RNN with Attention



Discuss: What are limitations of such sequence to sequence models? Hint: Think about runtime.

Language Modelling History



https://medium.com/@kirudang/language-model-history-before-and-after-transformer-the-ai-revolution-bedc7948a130

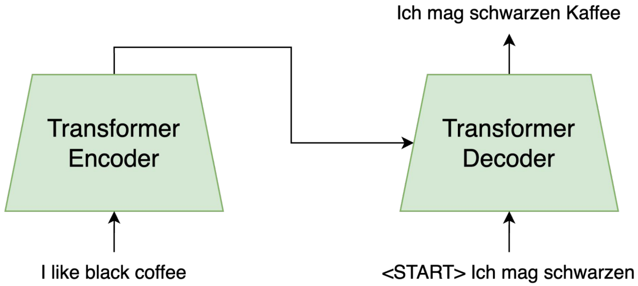
Self-Attention

A bat flew out of the dugout, startling the baseball player and making him drop his bat.

Cornell Bowers C·IS

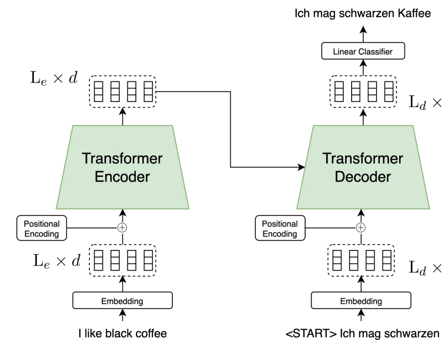
Transformer Architecture

Introduced for seq2seq tasks like Machine translation, summarization, question answering, etc.



Cornell Bowers C·IS

Transformer Architecture



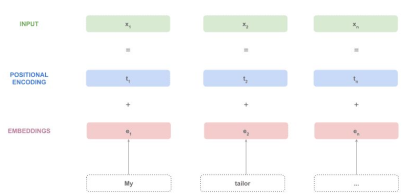
Cornell Bowers C·IS

Input Embeddings

- Replace tokens with continuous vectors
- Made up of two components:
  - token embeddings
  - positional encoding depends on position and dimension index as follows:

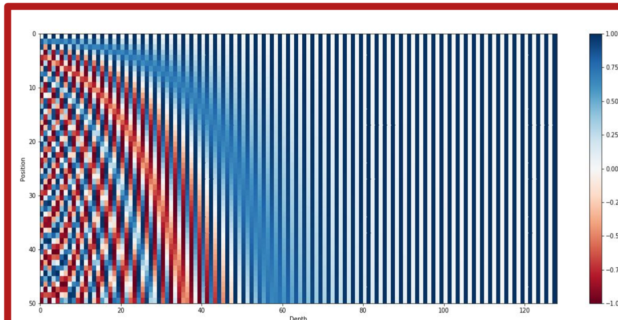
$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{2i/d_{\text{model}}}})$$

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

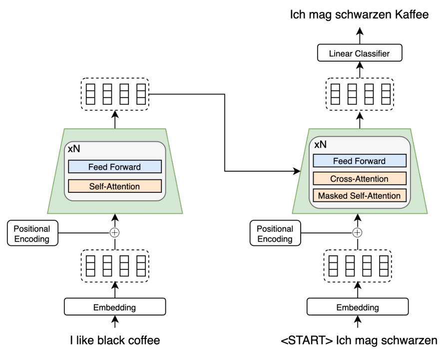


Cornell Bowers C·IS

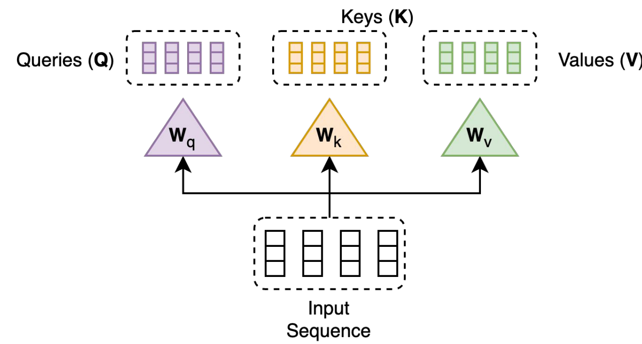
Example Positional Encoding



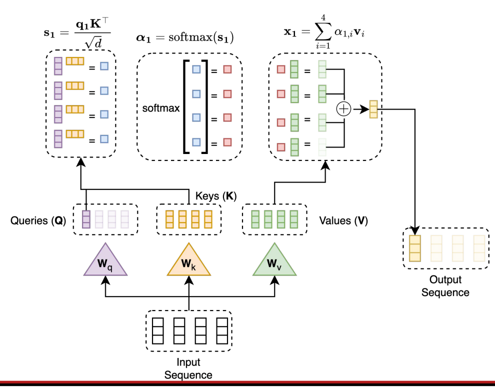
Transformer Architecture



Self-Attention



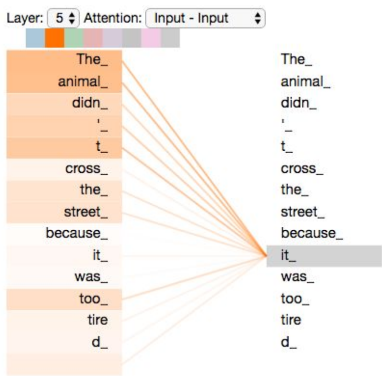
Self-Attention



Self-Attention: General Formula

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{d_k})V$$

Self-attention



Discuss:

- Q, K, V are all (n x d) matrices. Consider have an input of shape b x n x d.
- What is the shape of QK^T?
  - What does this matrix represent?
- What is the shape of the final output?
  - What does this matrix represent?

$$Attention(Q,K,V) = Softmax(\frac{QK^T}{d_k})V$$

Multi-Head Attention

What if I want to pay attention to different things at the same time!?

Content-based — This is my big red dog, Clifford.

Description-based — This is my big red dog, Clifford.

Reference-based — This is my big red dog, Clifford.

What's useful depends on the task. How do I pick what to do?

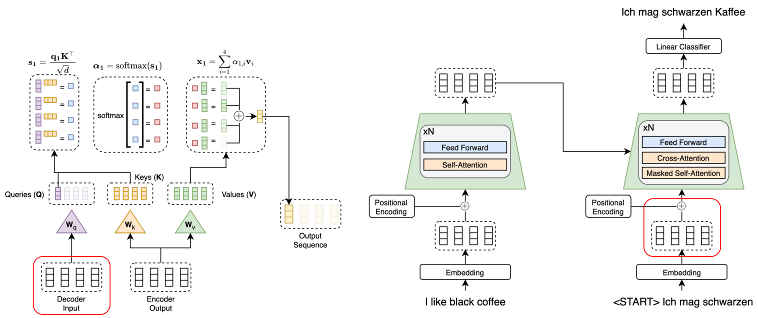
Multi-Head Attention

- The Scaled Dot-Product Attention attends to one or few entries in the input key-value pairs.
- Idea: apply Scaled Dot-Product Attention multiple times on the linearly transformed inputs.

$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

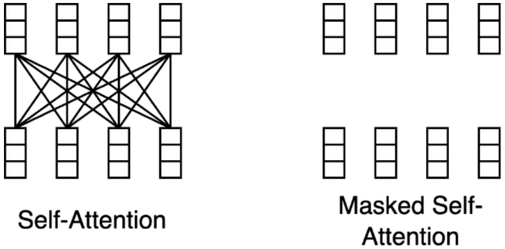
Cornell Bowers CIS

Cross-Attention



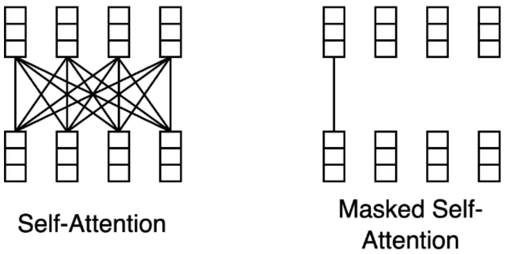
Cornell Bowers CIS

Self-Attention vs. Masked Self-Attention



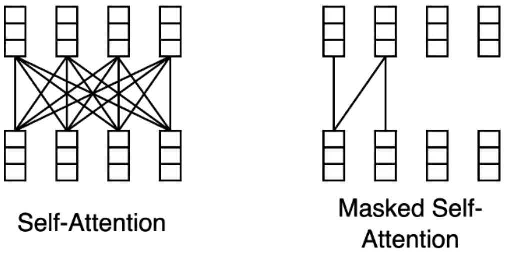
Cornell Bowers CIS

Self-Attention vs. Masked Self-Attention

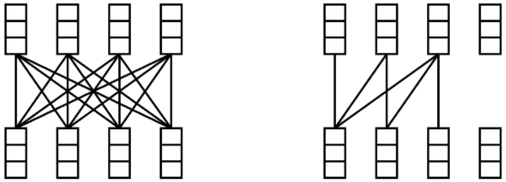


Cornell Bowers CIS

Self-Attention vs. Masked Self-Attention



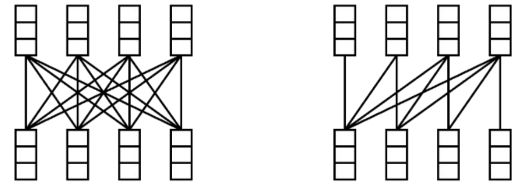
Self-Attention vs. Masked Self-Attention



Self-Attention

Masked Self-Attention

Self-Attention vs. Masked Self-Attention



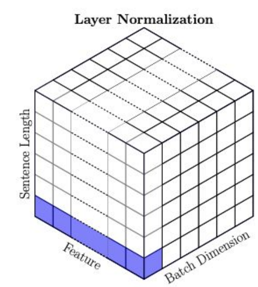
Self-Attention

Masked Self-Attention

Point-wise Feed-forward Networks

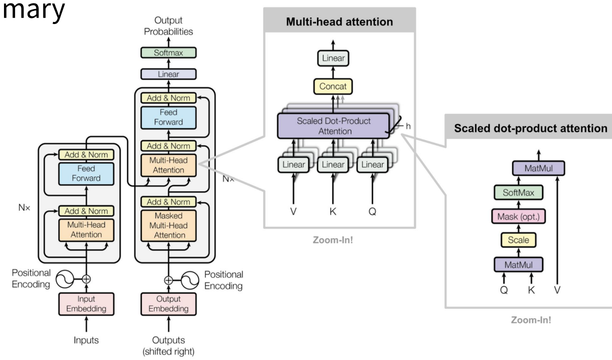
- Purpose
  - Applies non-linear transformations to the output of the attention layer
- Equation
  - $FFN(x) = max(0, xW_1 + b_1)W_2 + b_2$
    - where W and b are learned weights and biases
- These FFN is applied separately to each position

Layernorm



Cornell Bowers C·IS

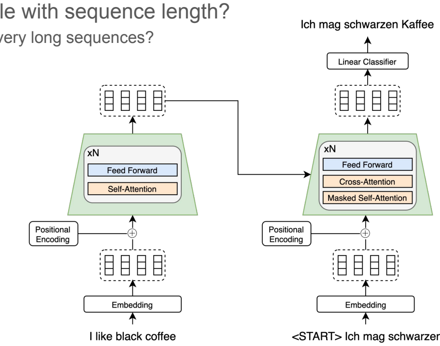
Summary



Cornell Bowers C·IS

Discuss:

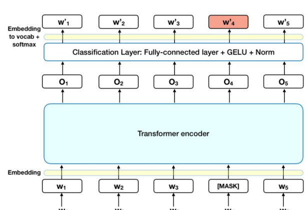
- How does the transformer scale with sequence length?
  - Any problems with applying it to very long sequences?



Cornell Bowers C·IS

BERT (Bidirectional Encoder Representations from Transformers)

- Bidirectional Context
- Pre-trained on the language, and then fine-tuned

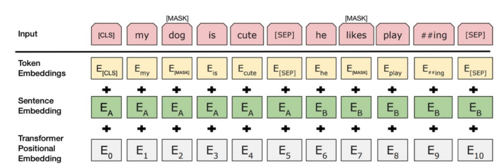


Cornell Bowers C·IS

BERT - Input Representation

Input:

- Use 30,000 WordPiece vocabulary on input.
- Each token is sum of three embeddings



Cornell Bowers C·IS

Training

- Masked Language Modelling
  - Mask out k% of the input words, and then predict the masked words
  - the man went to the store to [MASK] a [MASK] of milk
  - What can you use as a loss function?

- Next sentence prediction
  - To learn relationships between sentences, predict whether Sentence B is actual sentence that
  - proceeds Sentence A, or a random sentence

Sentence A = `The man went to the store.`
Sentence B = `He bought a gallon of milk.`
Label = `IsNextSentence`

Sentence A = `The man went to the store.`
Sentence B = `Penguins are flightless.`
Label = `NotNextSentence`

Cornell Bowers C·IS

Model Details

- Data: Wikipedia (2.5B words) + BookCorpus (800M words)
- Batch Size: 131,072 words (1024 sequences * 128 length or 256 sequences * 512 length)
- Training Time: 1M steps (~40 epochs)
- Optimizer: AdamW, 1e-4 learning rate, linear decay
- BERT-Base: 12-layer, 768-hidden, 12-head, 110M params
- BERT-Large: 24-layer, 1024-hidden, 16-head, 340M params
- Trained on 4x4 or 8x8 TPU slice for 4 days

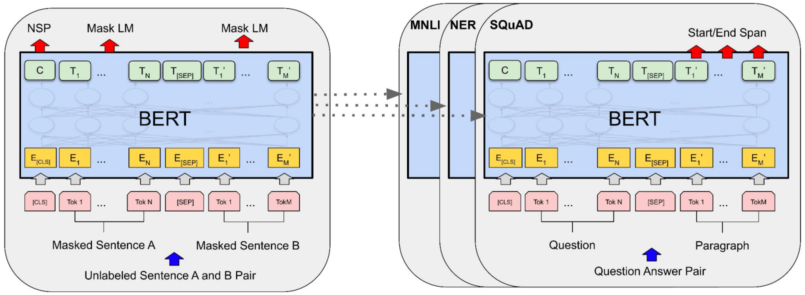
Cornell Bowers C·IS

Demo

https://huggingface.co/google-bert/bert-large-cased?text=Paris+is+the+capital+of+%5BMASK%5D.

Cornell Bowers C·IS

Pre-training to Fine-tuning Pipeline



Pre-training Fine-Tuning

Cornell Bowers CIS

System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

https://arxiv.org/pdf/1810.04805.pdf

Cornell Bowers CIS

Self-supervised Learning

- Labels are generated automatically, no human labeling process
- Benefits
  - Scales well
  - Cost-Efficient
  - Flexible
- Challenges
  - Larger datasets are required
  - More compute is necessary

Cornell Bowers CIS

Review

- LSTMs/GRUs are recurrent
- Self-attention can effectively replace recurrence in sequence-to-sequence models
- Transformers use self-attention and are parallelizable
- Pre-training using self-supervised learning help train large models that learn very good representations