Washington University in St. Louis
James McKelvey School of Engineering

CSE 561A: Large Language Models

Spring 2024

Lecture 2: Language Model Architectures and Pre-training

Content

- **Transformers: Self-Attention**
- Different Architectures of Pre-trained Language Models
  - Decoder-Only Models (GPT)
  - Encoder-Only Models (BERT)
  - Encoder-Decoder Models (T5, BART)

Recap: Context-Free Embedding

- 1) Each word is mapped to only one vector regardless of its context!
  - E.g. “bank” is a polysemy, but only has one representation

“Open a **bank** account”

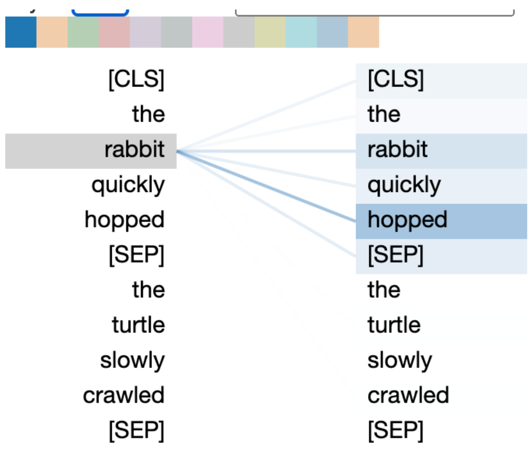
“On the river **bank**”

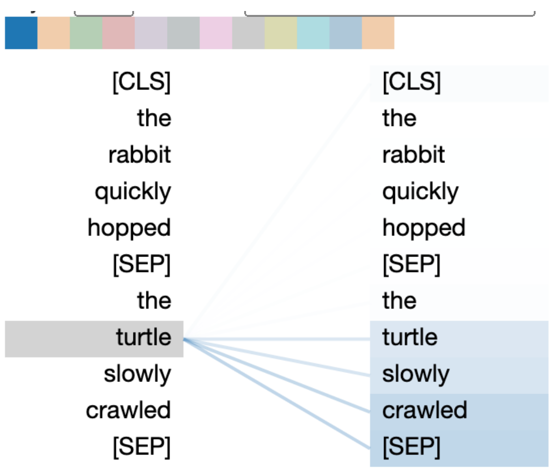
Share representation

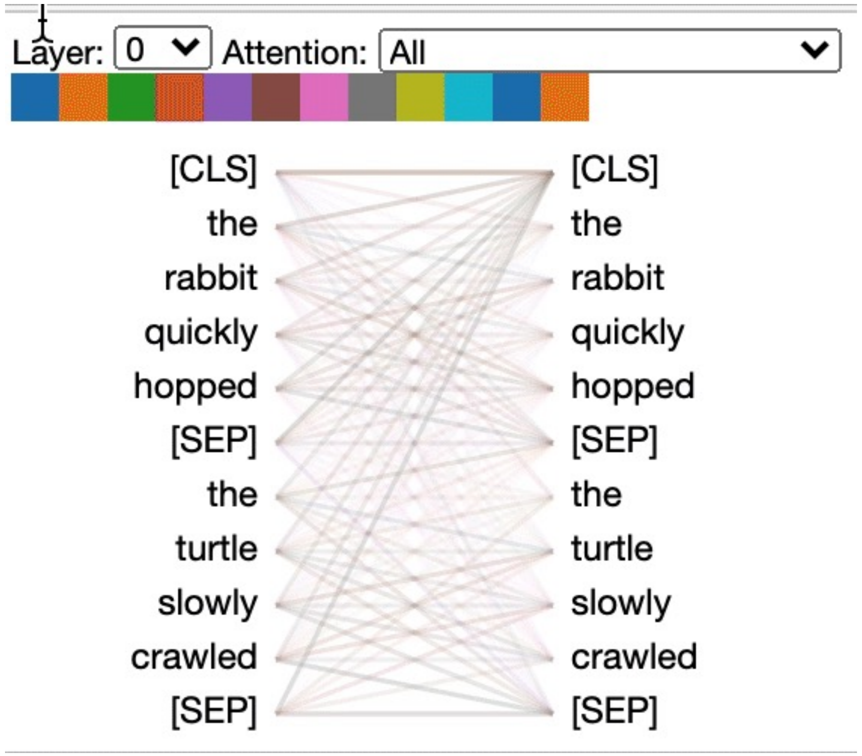
- 2) It does not consider the order of words
- 3) It treats the words in the context window equally
- Solution: We need **contextualized** text representations!
  - Injecting context information into words via advanced model architectures

Attention is all you need (Transformer)

- Self-Attention: Each token attends to every other token in the sentence, but with different weights
- Demo: https://github.com/jessevig/bertviz

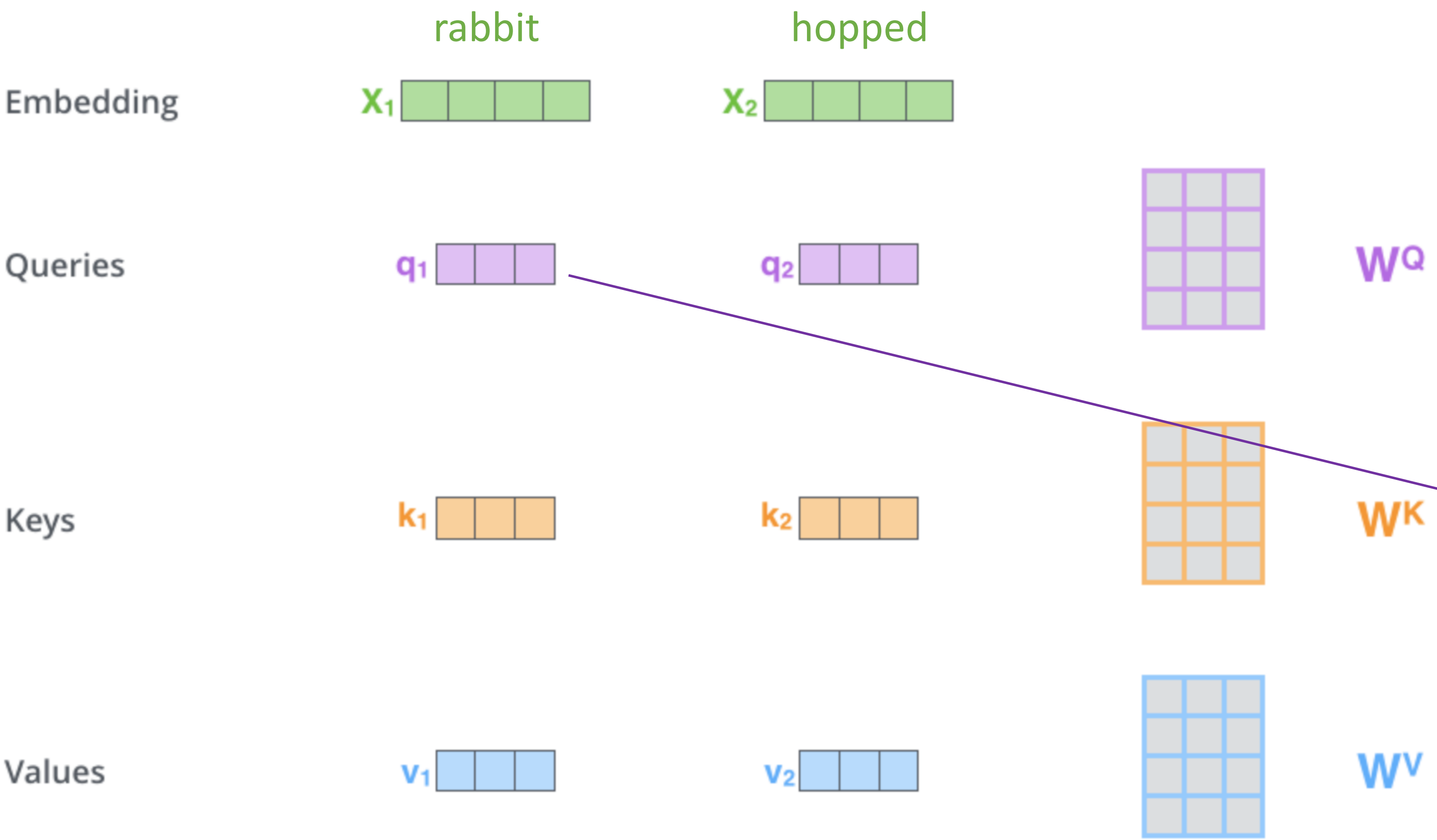




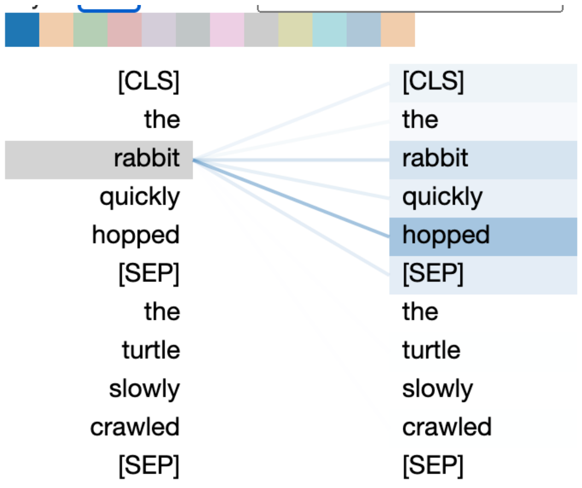


Self-Attention

- Each word is represented as a query, key and value vector. The vectors are obtained from the input embeddings multiplied by a weight matrix.



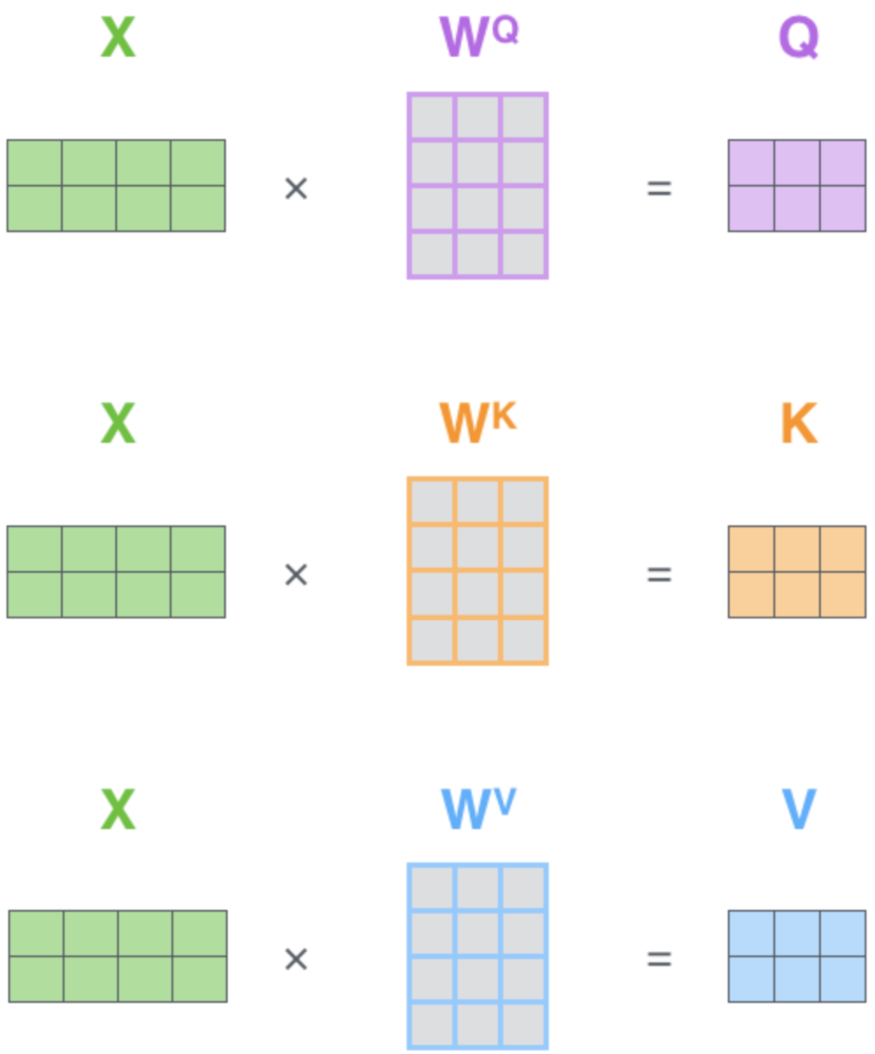
Eg., Represent “rabbit” with its context:

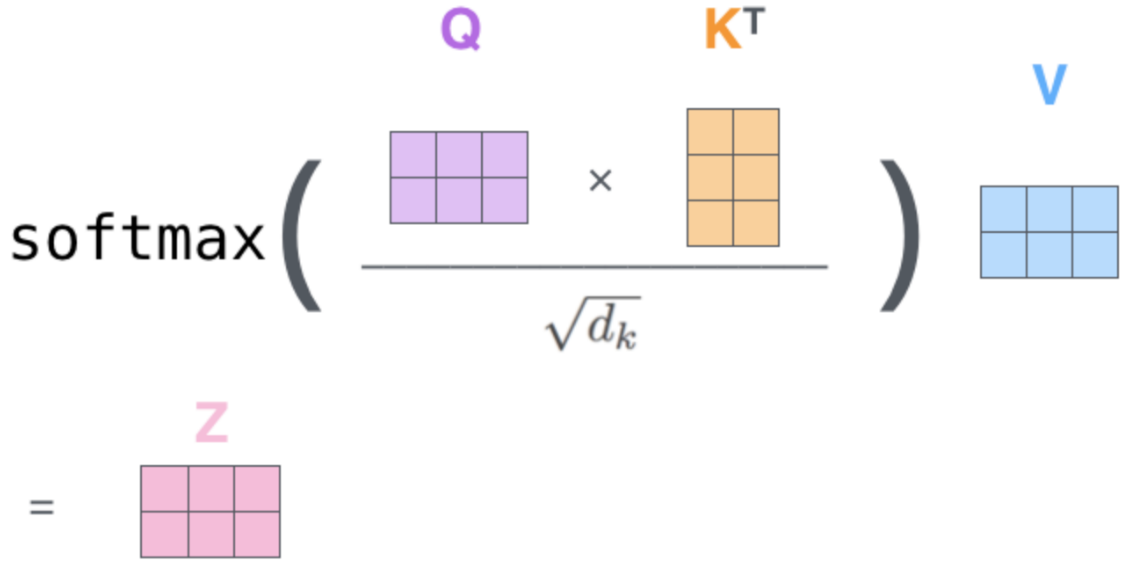


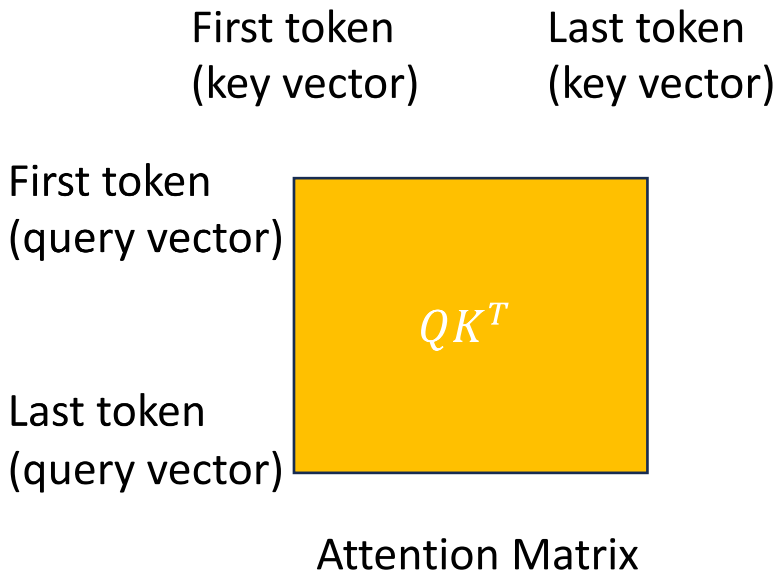
$$\sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i$$

Attention: normalized dot product of q and k_i

Self-Attention: Matrix Calculation





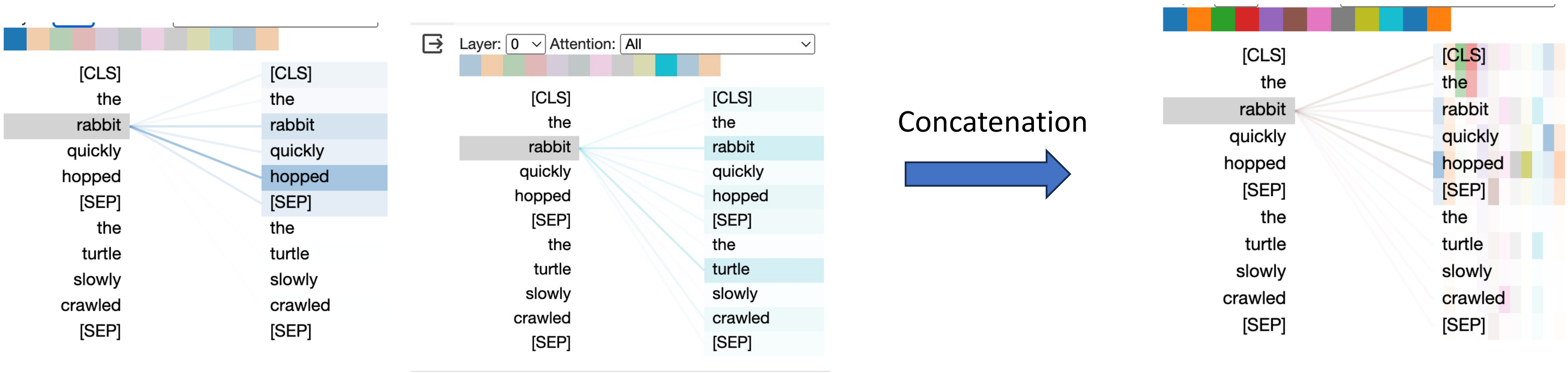


Multi-Head Attention

- Input: Multiple Independent sets of query, key, value matrices
- Output: Concatenate the outputs of attention heads
- Advantage: Each attention head focus on one subspace

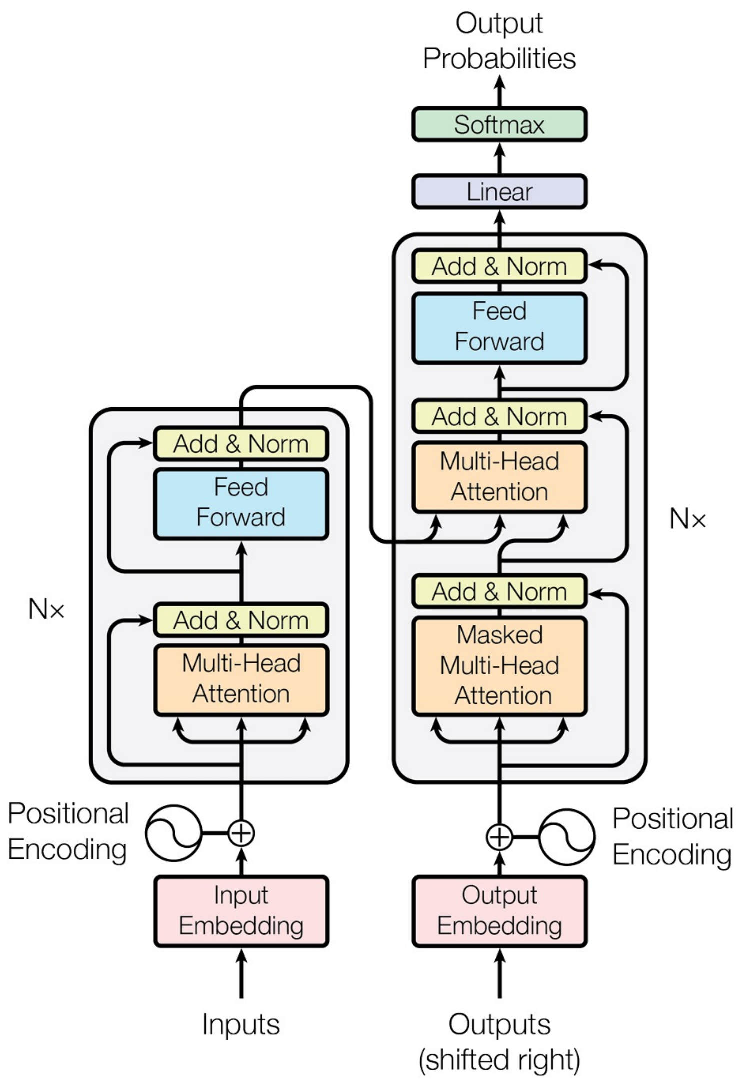
$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, \ldots, \mathrm{head_h})W^O$$

$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



Transformer Model Architecture

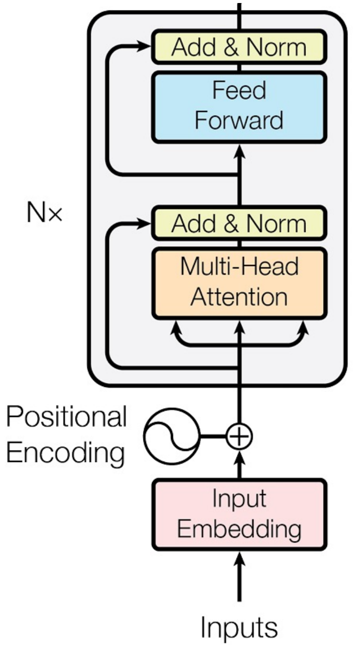
- Input Embedding
- Positional Encoding
- 12 Transformer layers
  - 6 encoder layers: text understanding
  - 6 decoder layers: text generation
- Output: Linear + SoftMax layer for next word prediction



Encoder Layer

- Multi-head attention layer captures information from different subspaces at different positions

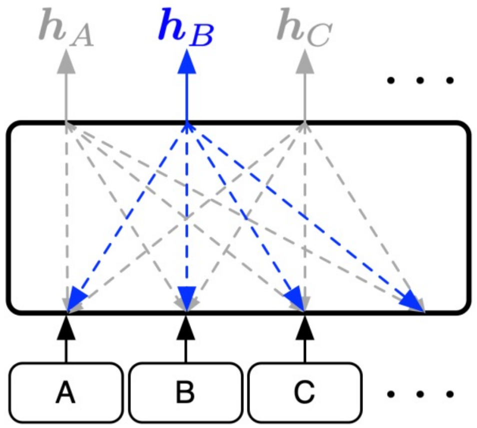
$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$



- Feed-forward layer is applied to each token position

without interaction with other positions

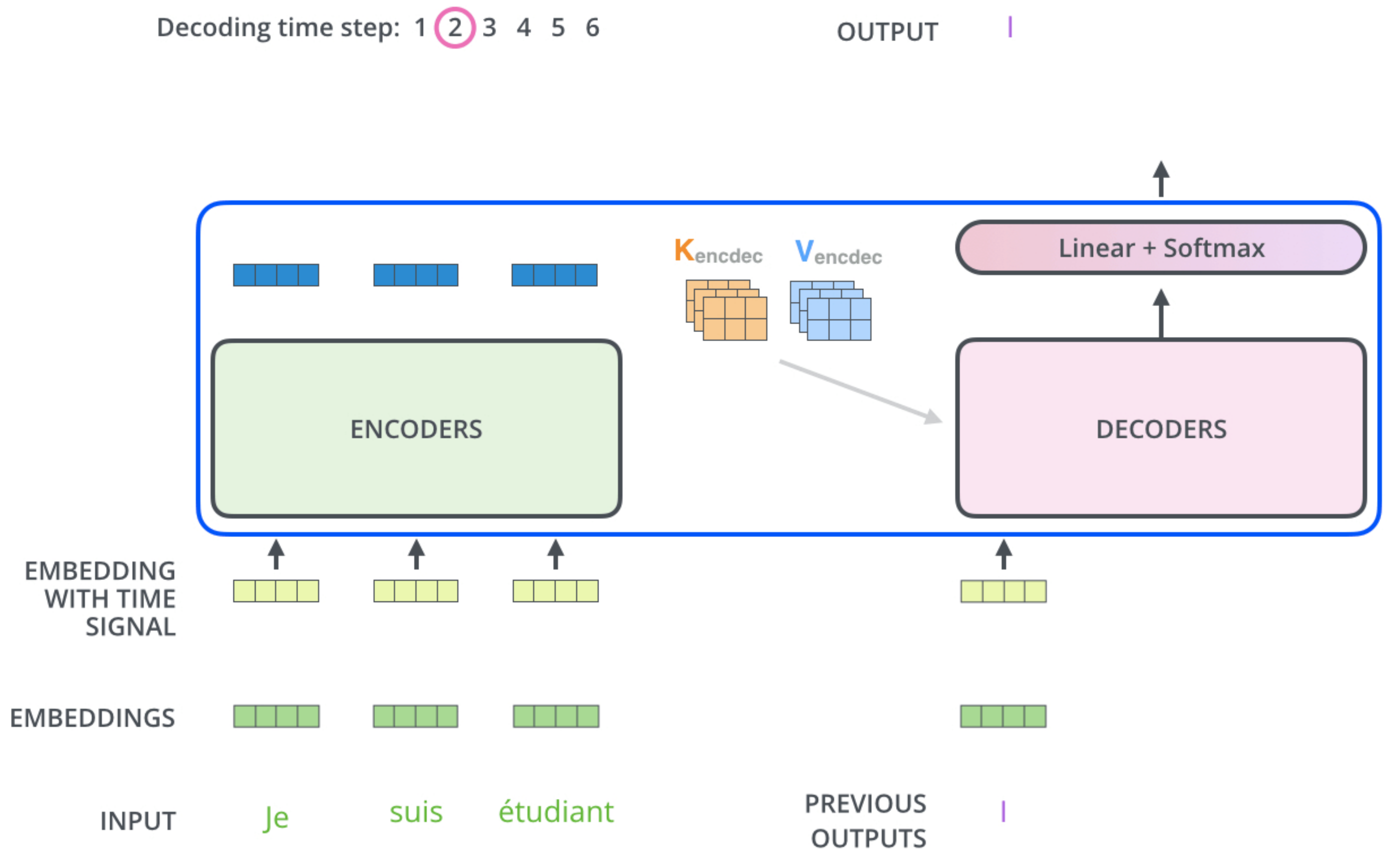
$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$



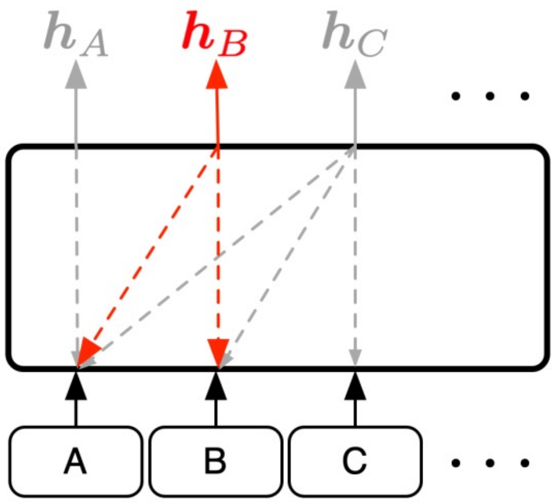
- **Bidirectional attention**:
- Every token attends to all tokens

Decoder Layer

- Demo from https://jalammar.github.io/illustrated-transformer/

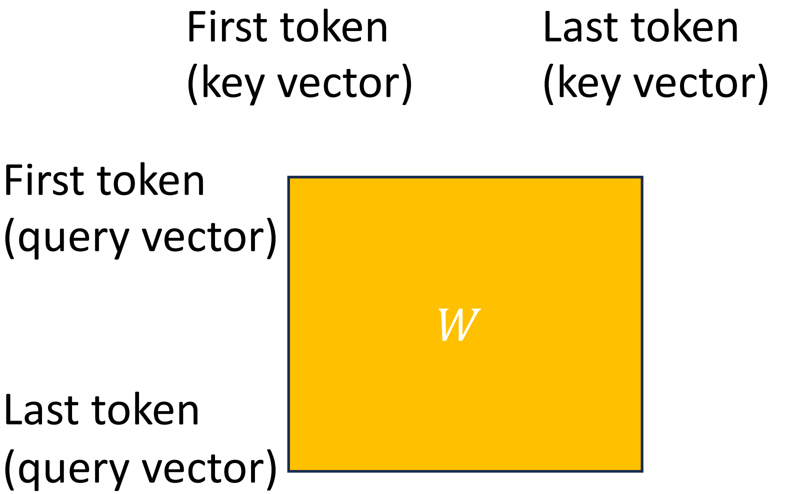


Decoder Layer

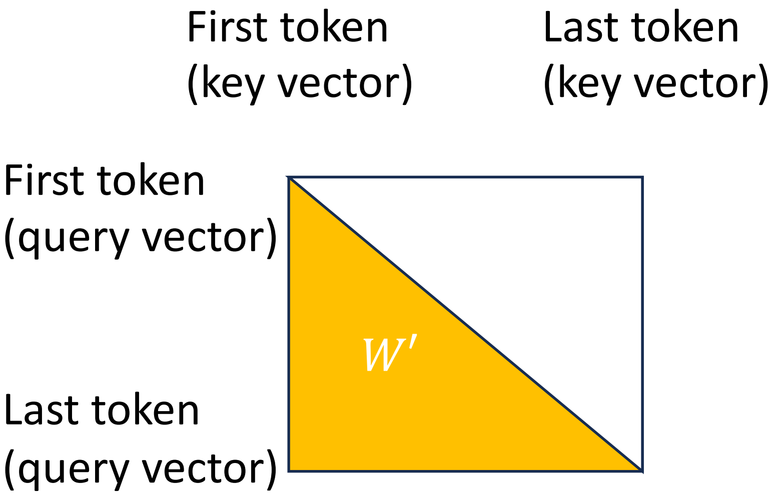


- **Unidirectional Self-Attention**:
- Every token attends to its previous tokens

- Attention Matrix



Attention Matrix (Encoder)



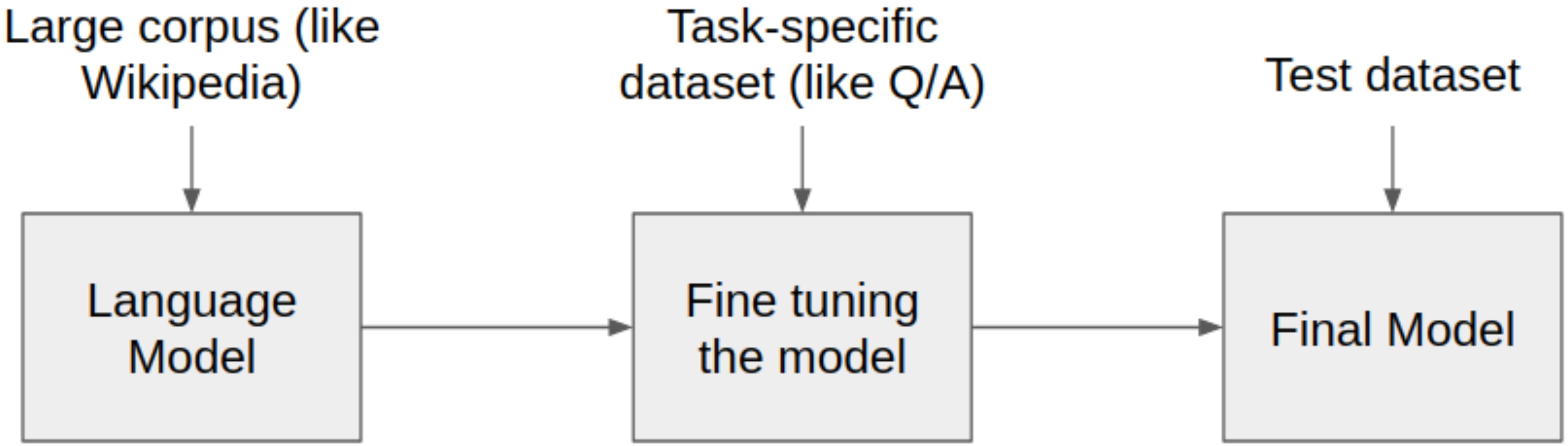
Attention Matrix (Decoder)

Content

- Transformers: Self-Attention
- **Different Architectures of Pre-trained Language Models**
    - Decoder-Only Models (GPT)
    - Encoder-Only Models (BERT)
    - Encoder-Decoder Models (T5, BART)

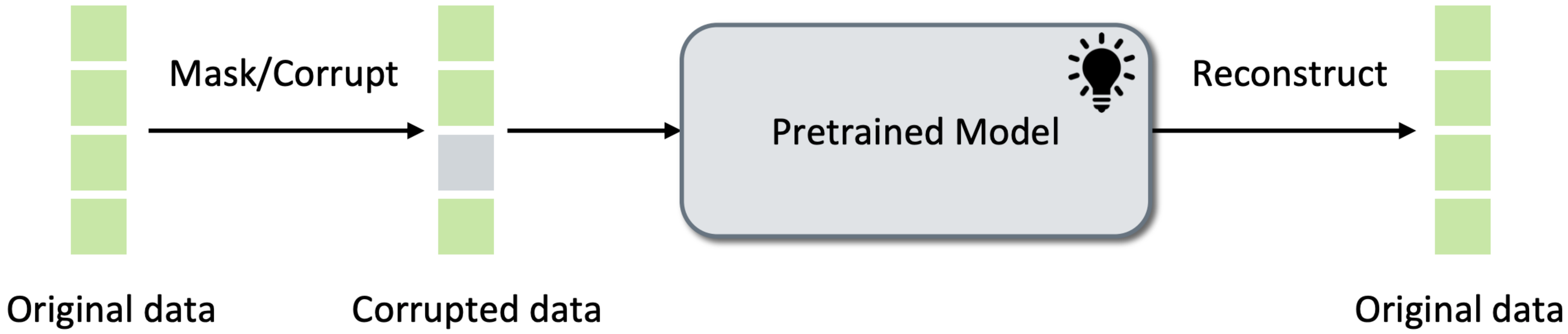
Pretrain-Finetune Paradigm

- "Pretraining": Train deep language models (usually Transformer models) via **self-supervised** objectives on **large-scale general-domain corpora**
- "Fine-tuning": Adapt the pretrained language models (PLMs) to downstream tasks by further training on task-specific data
- The power of PLMs: Encode generic linguistic features and knowledge learned through large-scale pretraining, which can be effectively transferred to the target applications



Overview of Pretraining

- Self-supervised learning
- Make a part of the input unknown to the model
- Let the model predict that unknown part based on the known part



Different Architectures for PLMs

- **Decoder-Only (Unidirectional) PLM** (e.g., GPT): Predict the next token based on previous tokens, usually used for **language generation tasks**
- **Encoder-Only (Bidirectional) PLM** (e.g., BERT, XLNet, ELECTRA): Predict masked/corrupted tokens based on all other (uncorrupted) tokens, usually used for **language understanding/classification tasks**
- **Encoder-Decoder (Sequence-to-Sequence) PLM** (e.g., T5, BART): Generate output sequences given masked/corrupted input sequences, can be used for both **language understanding and generation tasks**

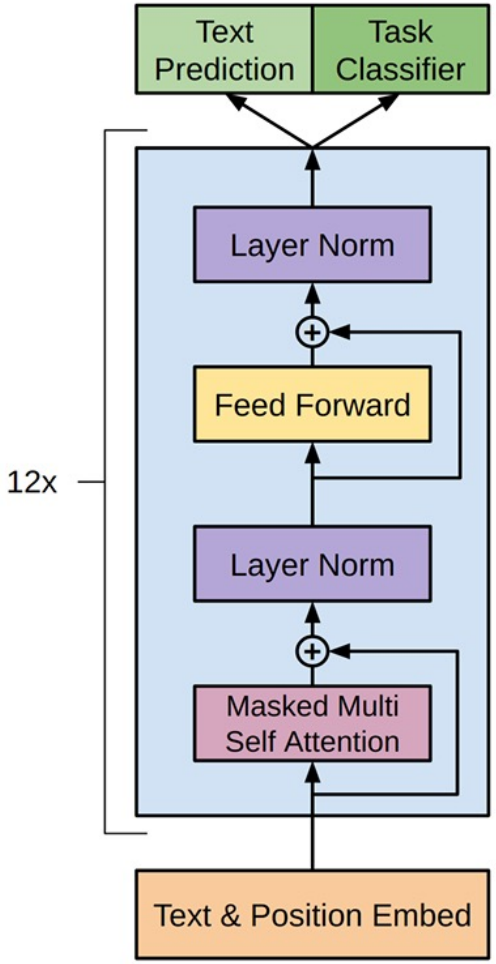
Decoder Pretraining (GPT)

- Model Architecture: A multi-layer transformer decoder
- Leverage unidirectional context (usually left-to-right) for next token prediction (i.e., language modeling)

k previous tokens as context

$$\mathcal{L}_{\mathrm{LM}} = -\sum_i \log p(x_i \mid \boxed{x_{i-k}, \ldots, x_{i-1}})$$

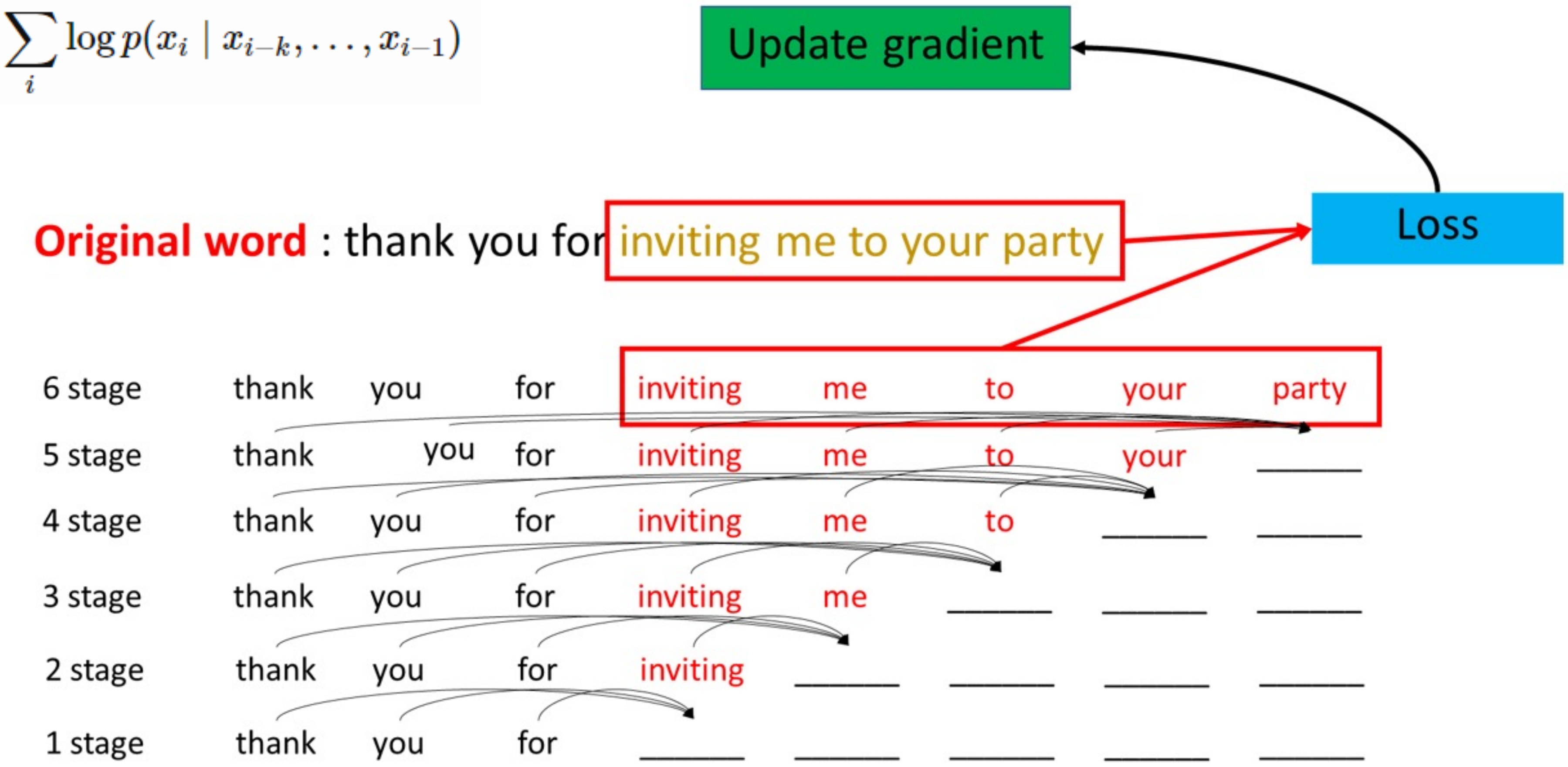
- The Transformer uses **unidirectional** attention masks (i.e., every token can only attend to previous tokens)
- Decoder architecture is the prominent choice in large language models



[1] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI blog.

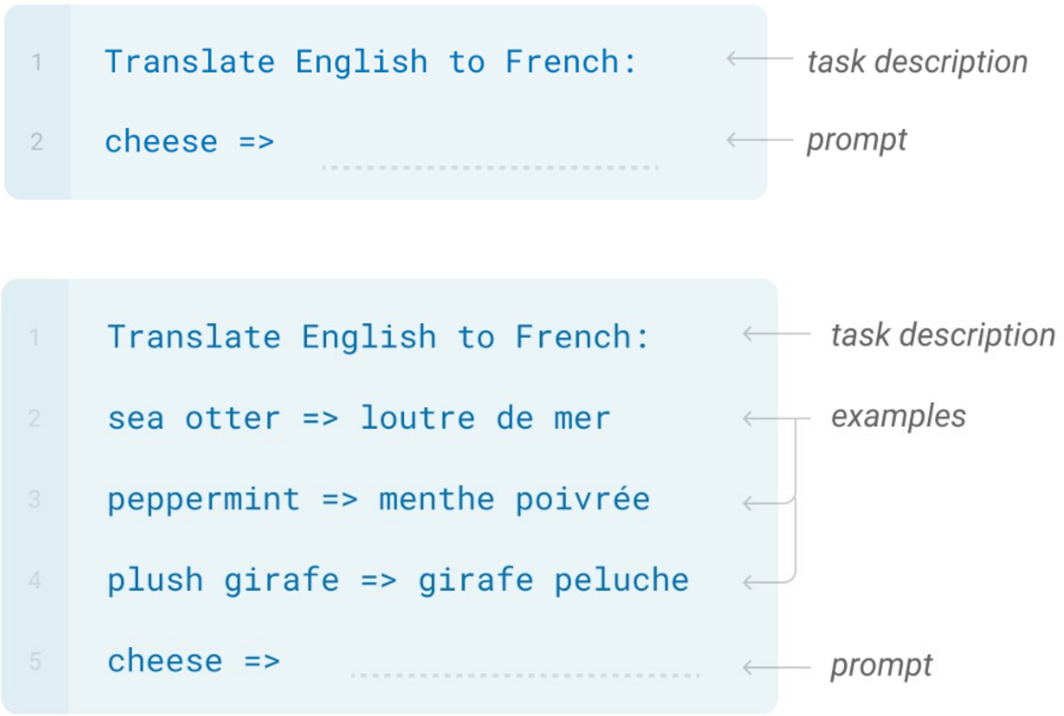
Decoder Pretraining

$$\mathcal{L}_{\mathrm{LM}} = -\sum_i \log p(x_i \mid x_{i-k}, \ldots, x_{i-1})$$



Usage of Decoder Models

- Question Answering

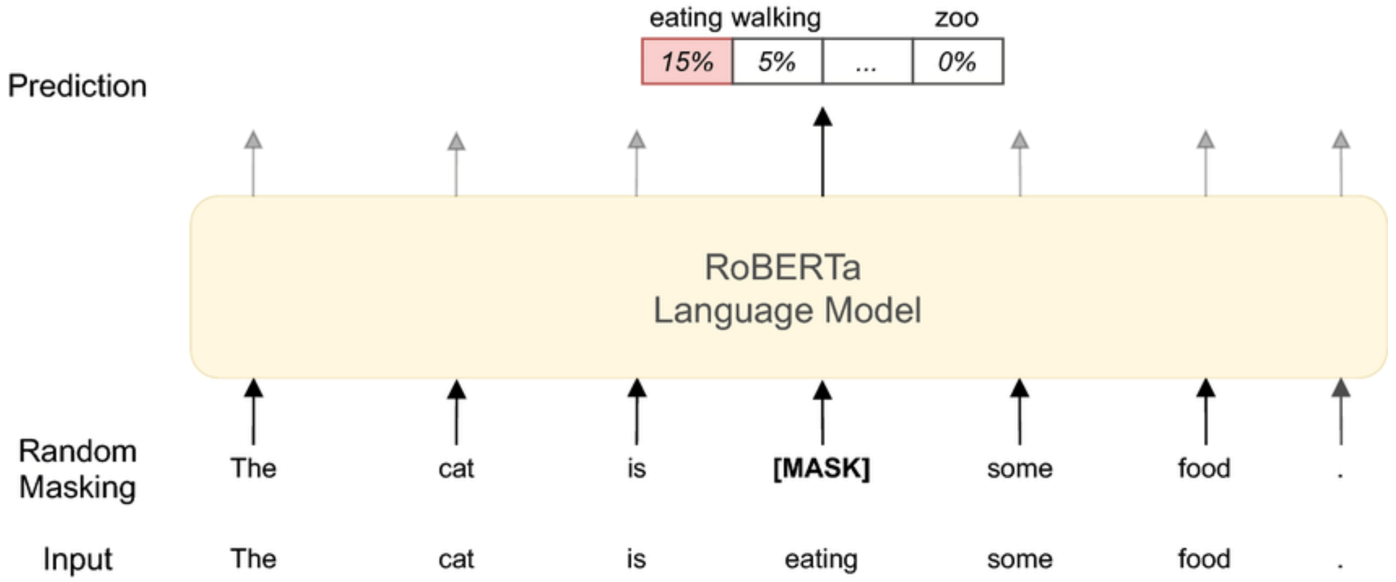


Content

- Transformers (continued)
- Different Architectures of Pre-trained Language Models
  - Decoder-Only Models (GPT)
  - **Encoder-Only Models (BERT)**
  - Encoder-Decoder Models (BART, T5)

BERT Model Architecture

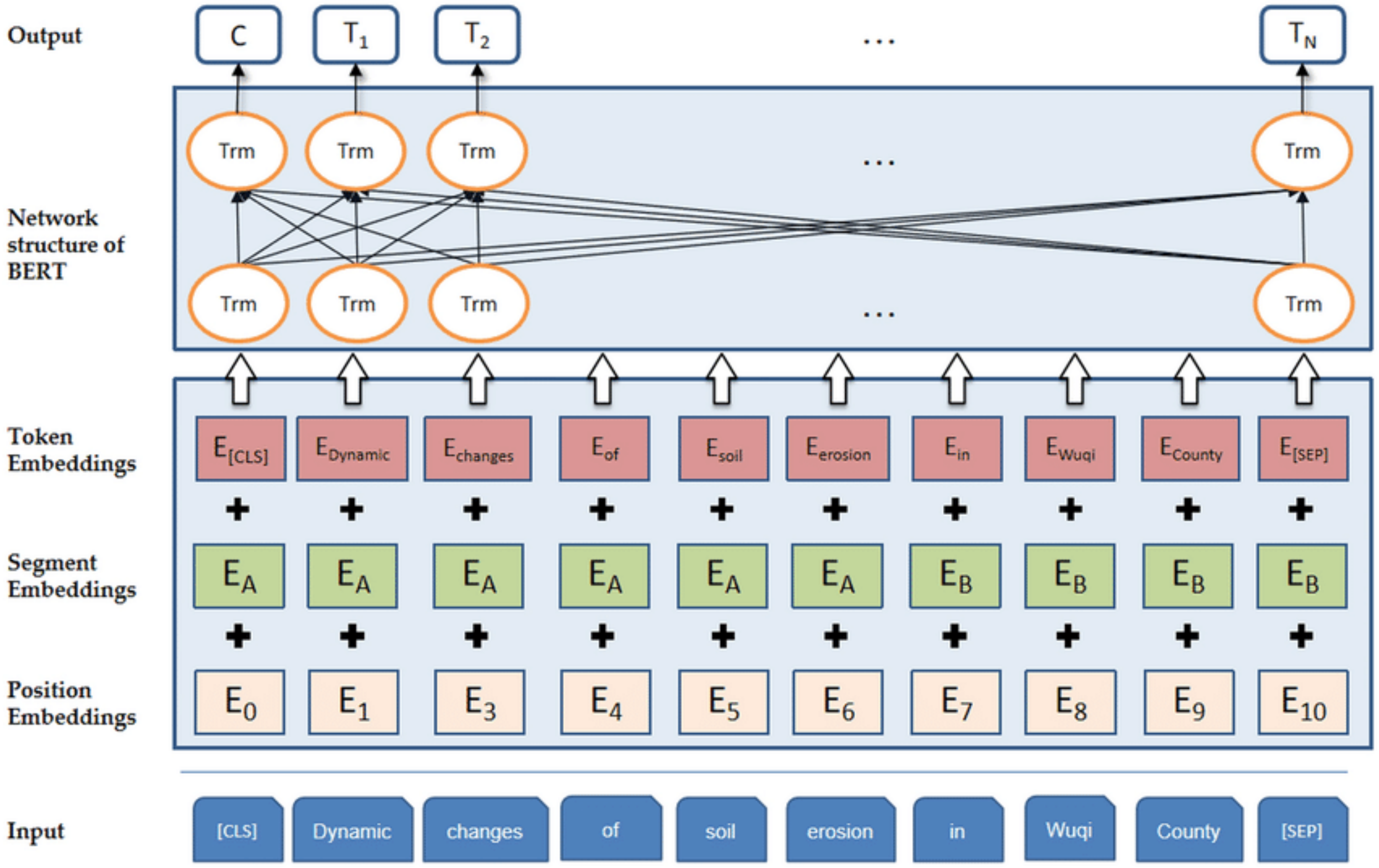
- Pre-training objectives
  - Masked language modeling (with bidirectional attention) + Next Sentence Prediction
  - 15% of tokens are randomly corrupted (masked) for model prediction



[1] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Devlin et al. NAACL’19.

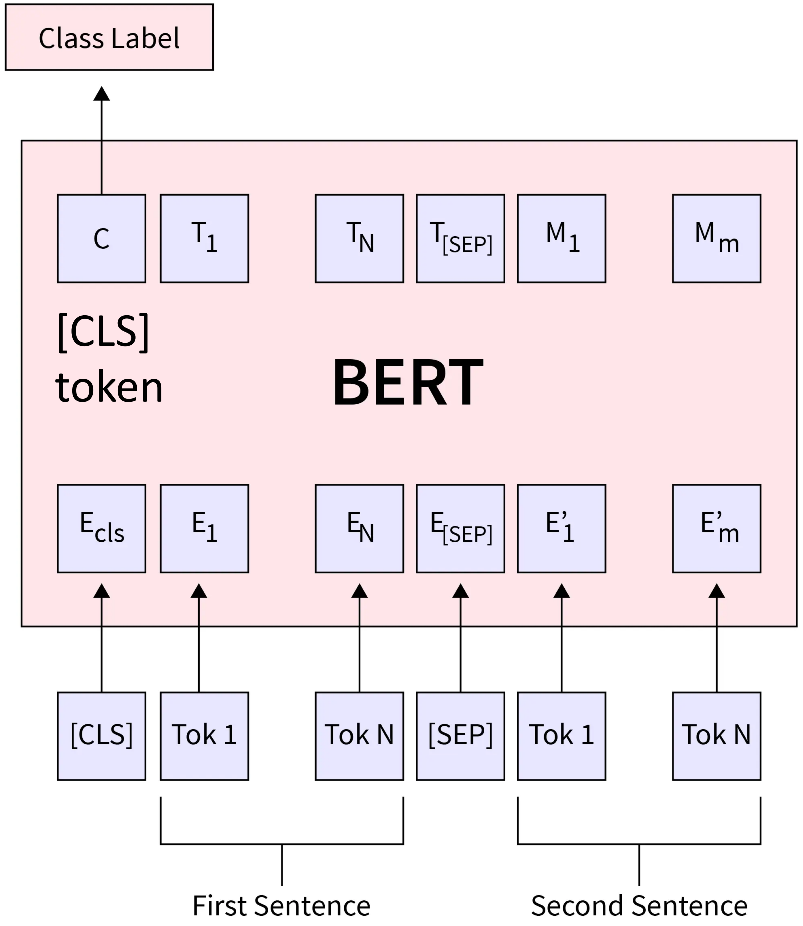
BERT Model Architecture

- Bidirectional attention: each token can attend to its left and right context for self-attention



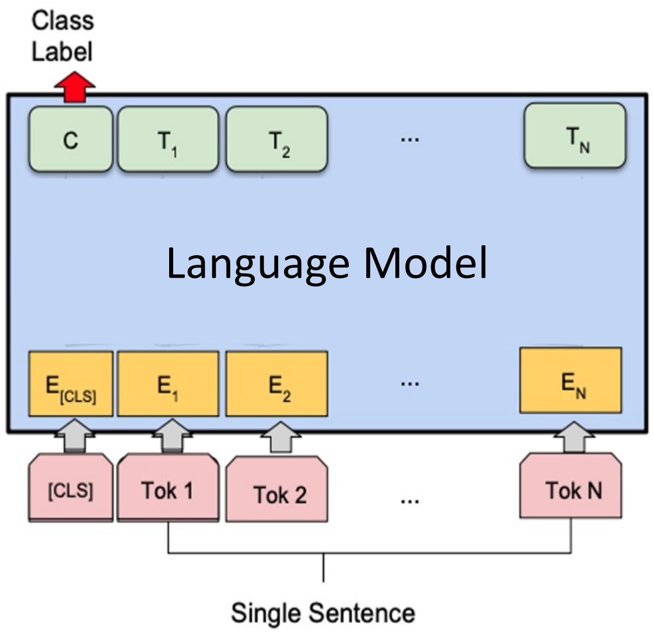
Next Sentence Prediction

- Next Sentence Prediction (NSP)
  - Predict whether Sentence B is the next sentence of Sentence A.
  - Positive samples: two contiguous sentences in the corpus.
  - Negative samples: sample another sentence for sentence A.
  - Class Labels: <is_next, not_next>



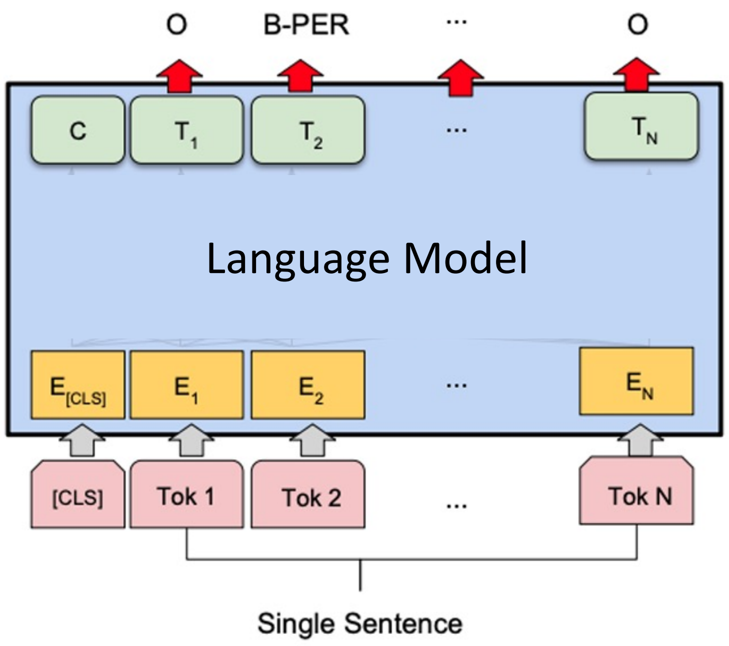
Usage of Encoder Models (I)

- Sentence classification tasks
  - Text Classification Tasks
    - Input: The bike is too small and I want to return it.
    - Output: <refund, return, check_status>
  - Sentiment Analysis
    - Input: The restaurant is crowded and I waited my food for 30 minutes!
    - Output: <positive, negative>



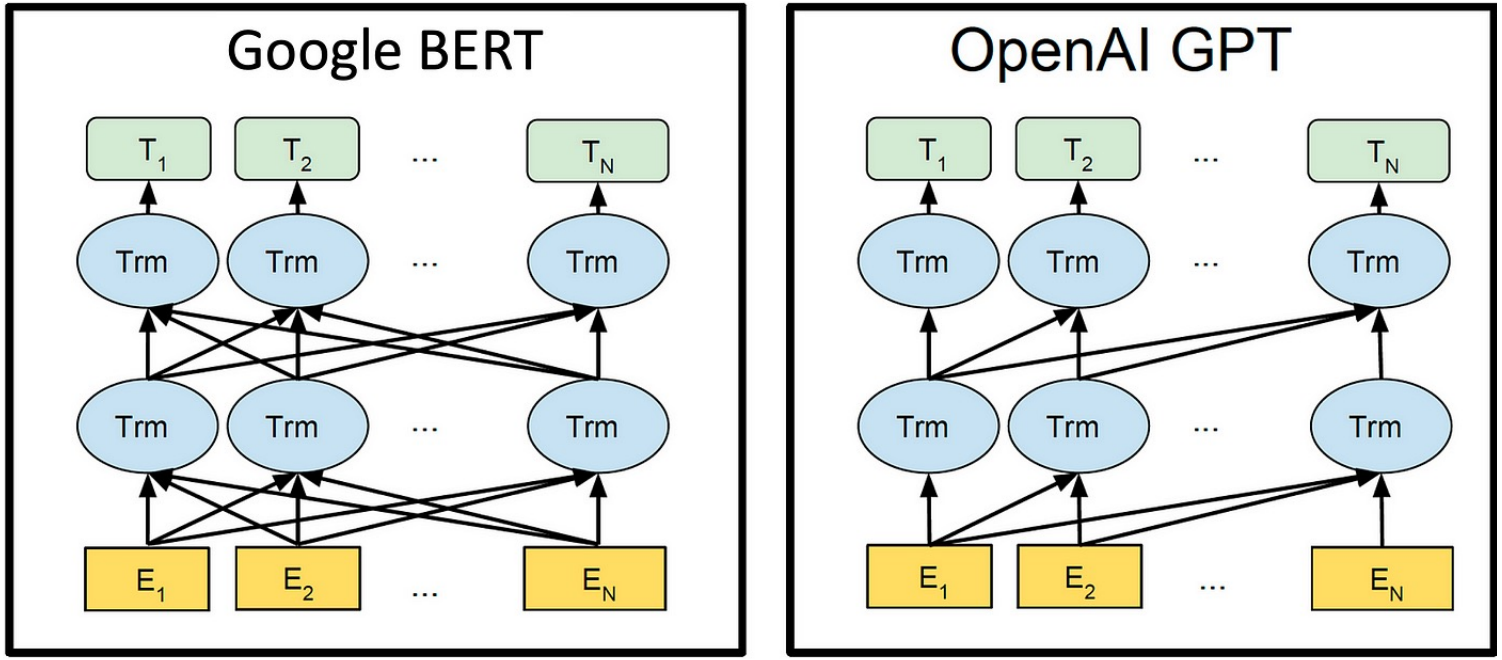
Usage of Encoder Models (II)

- Token-level tasks
  - Named Entity Recognition
  - Input: St. Louis is located in the state of Missouri .
  - Output: <Begin-Location> <Inside-location> O O O O O O <Begin-Location> O



Comparison with GPT Model

- Training objective: MLM prediction vs. left-to-right token prediction



Performance comparison between BERT and GPT-1

- GLUE Benchmark for natural language understanding
- BERT is better at language understanding

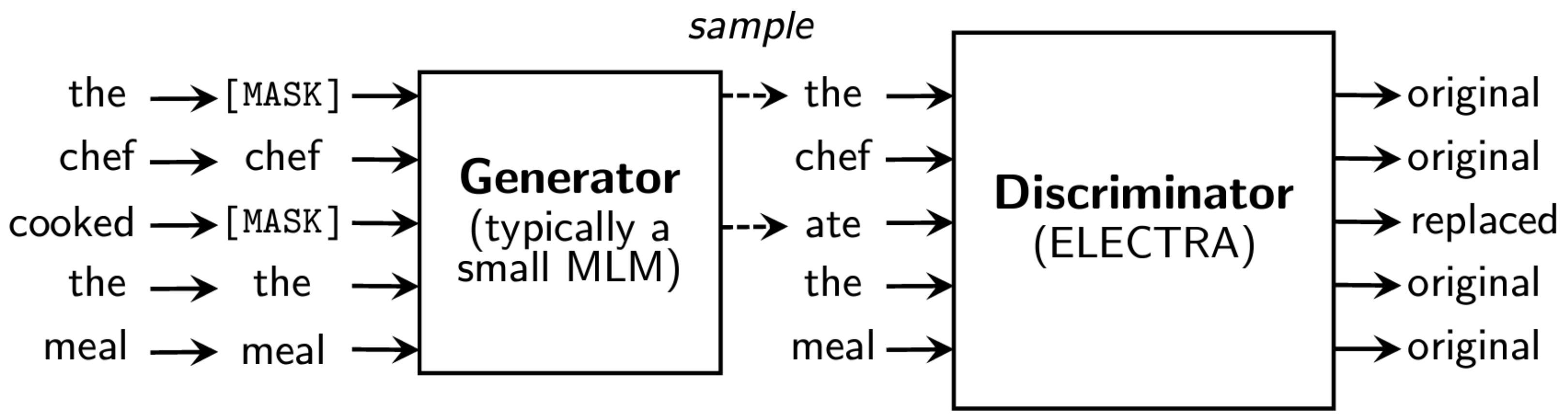
System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	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**

Variants of BERT Model

- RoBERTa (RoBERTa: A Robustly Optimized BERT Pretraining Approach. Liu et al. 2019)
  - Training the model longer on more data with bigger batches
  - Remove the next sentence prediction objective
  - Dynamically change the [MASK] patterns in each epoch

Variants of BERT Model

- ELECTRA (ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. Clark et al. 2020)
  - Replaced token detection by corrupting text sequences with an auxiliary MLM
  - Works better than BERT because the input text for ELECTRA does not contain [MASK] tokens (no discrepancy between training and test data)

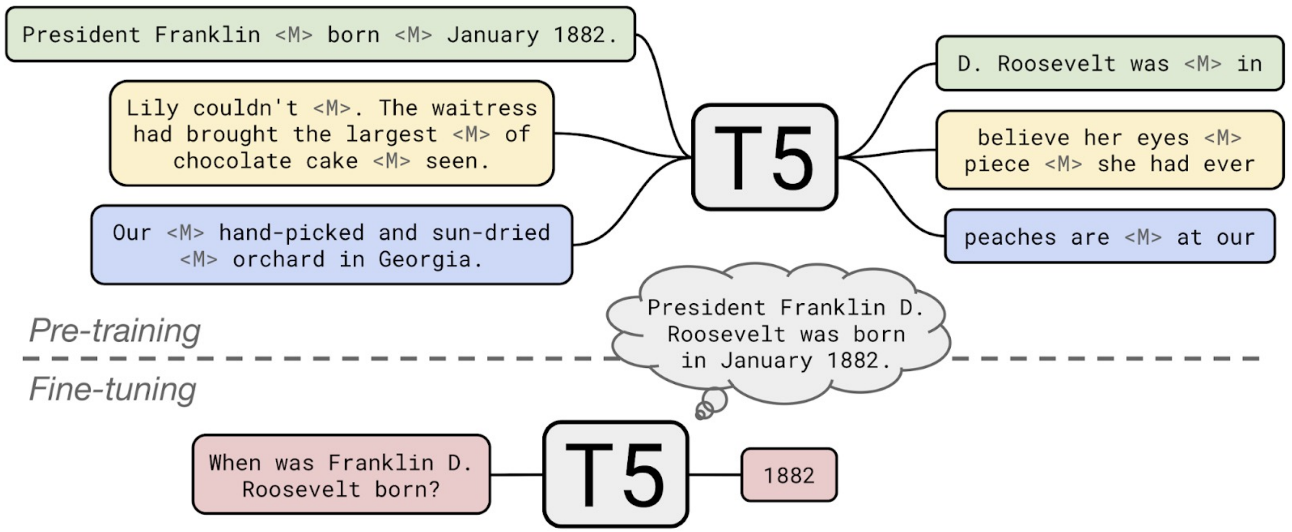


Content

- Transformers
- Different Architectures of Pre-trained Language Models
  - Decoder-Only Models (GPT)
  - Encoder-Only Models (BERT)
  - **Encoder-Decoder Models (T5, BART)**

T5 Model

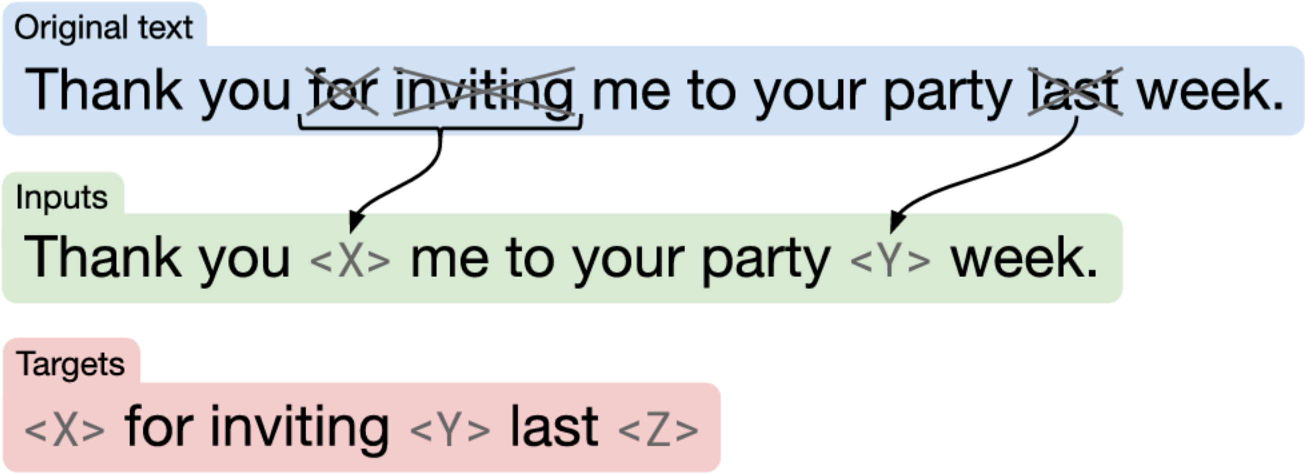
- How to predict a span of masked tokens within a sentence?
- BERT model requires the number of [MASK] token to be given in prior, while GPT models are causal left-to-right models
- T5: **T**ext-**t**o-**T**ext **T**ransfer **T**ransformer (parameters: 60M~11B)



Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. JMLR.

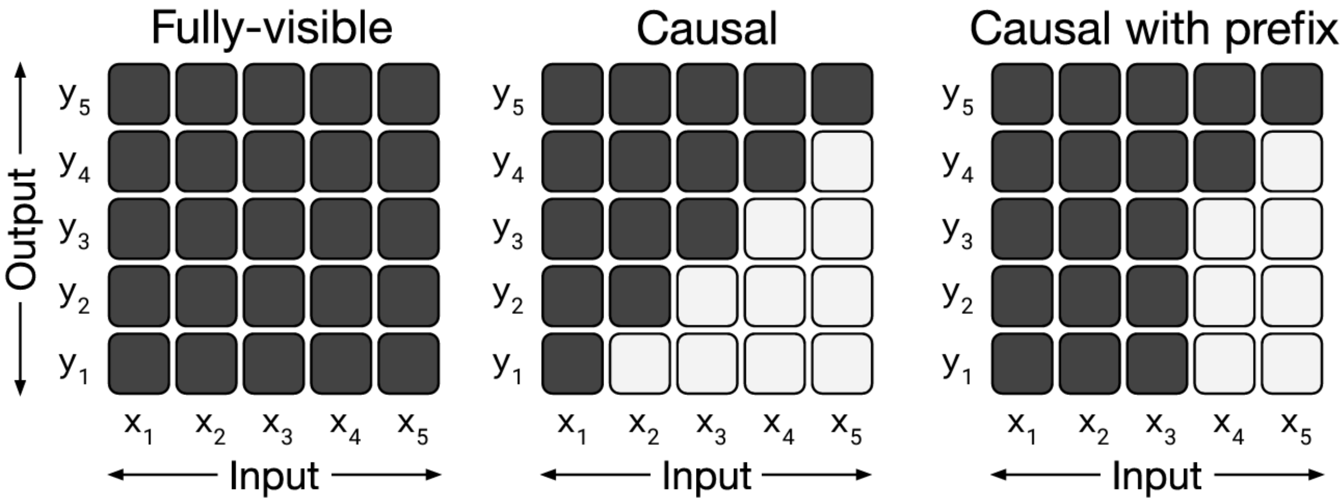
Training of T5 Model

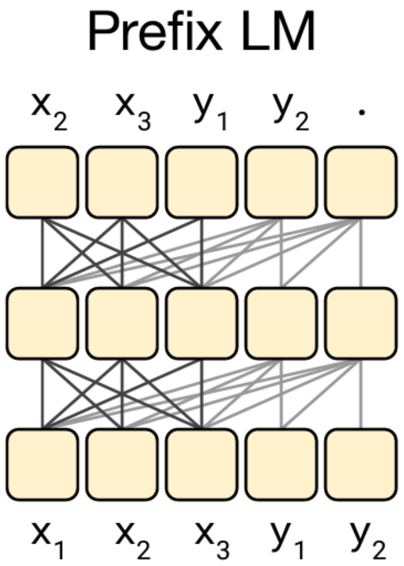
- Pretraining: Mask out spans of texts; generate the original spans
- Fine-Tuning: Convert every task into a sequence-to-sequence generation problem
- Text-to-Text: Uncertain number of tokens in the input, and uncertain number of tokens in the output



T5 Attention

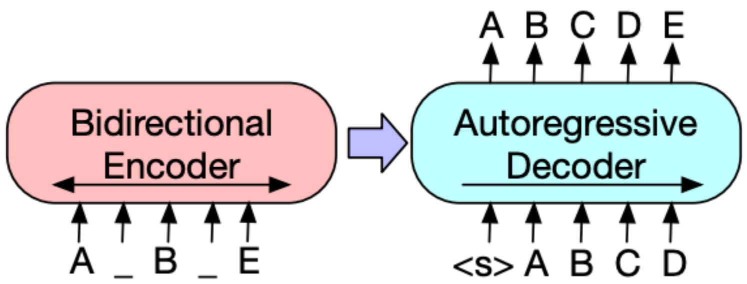
- A “fully-visible” attention mechanism is placed at the input sequence.
  - Input Sequence:
    - translate English to German : That is good . target :
  - Target Output:
    - Das ist gut .



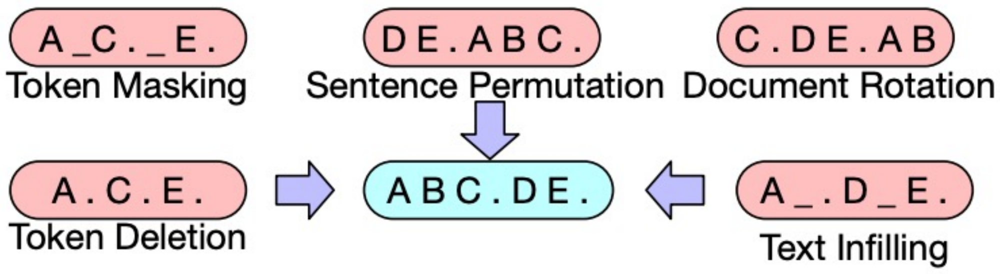


BART Model

- BART: Denoising autoencoder for pretraining sequence-to-sequence models
- Pretraining: Apply a series of noising schemes (e.g., masks, deletions, permutations…) to input sequences and train the model to recover the original sequences



BART architecture



BART pretraining objectives

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... & Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. ACL.

Performance Comparison

- Comparable to encoder models on language understanding tasks
- Better performance on language generation tasks

	SQuAD 1.1 EM/F1	**SQuAD 2.0** EM/F1	**MNLI** m/mm	**SST** Acc	**QQP** Acc	**QNLI** Acc	**STS-B** Acc	**RTE** Acc	**MRPC** Acc	**CoLA** Mcc
BERT	84.1/90.9	79.0/81.8	86.6/-	93.2	91.3	92.3	90.0	70.4	88.0	60.6
UniLM	-/-	80.5/83.4	87.0/85.9	94.5	-	92.7	-	70.9	-	61.1
XLNet	**89.0**/94.5	86.1/88.8	89.8/-	95.6	91.8	93.9	91.8	83.8	89.2	63.6
RoBERTa	88.9/**94.6**	**86.5/89.4**	**90.2/90.2**	96.4	92.2	94.7	**92.4**	86.6	**90.9**	**68.0**
BART	88.8/**94.6**	86.1/89.2	89.9/90.1	**96.6**	**92.5**	**94.9**	91.2	**87.0**	90.4	62.8

	CNN/DailyMail			**XSum**		
	R1	R2	RL	R1	R2	RL
Lead-3	40.42	17.62	36.67	16.30	1.60	11.95
PTGEN (See et al., 2017)	36.44	15.66	33.42	29.70	9.21	23.24
PTGEN+COV (See et al., 2017)	39.53	17.28	36.38	28.10	8.02	21.72
UniLM	43.33	20.21	40.51	-	-	-
BERTSUMABS (Liu & Lapata, 2019)	41.72	19.39	38.76	38.76	16.33	31.15
BERTSUMEXTABS (Liu & Lapata, 2019)	42.13	19.60	39.18	38.81	16.50	31.27
BART	**44.16**	**21.28**	**40.90**	**45.14**	**22.27**	**37.25**

Scaling up Language Models

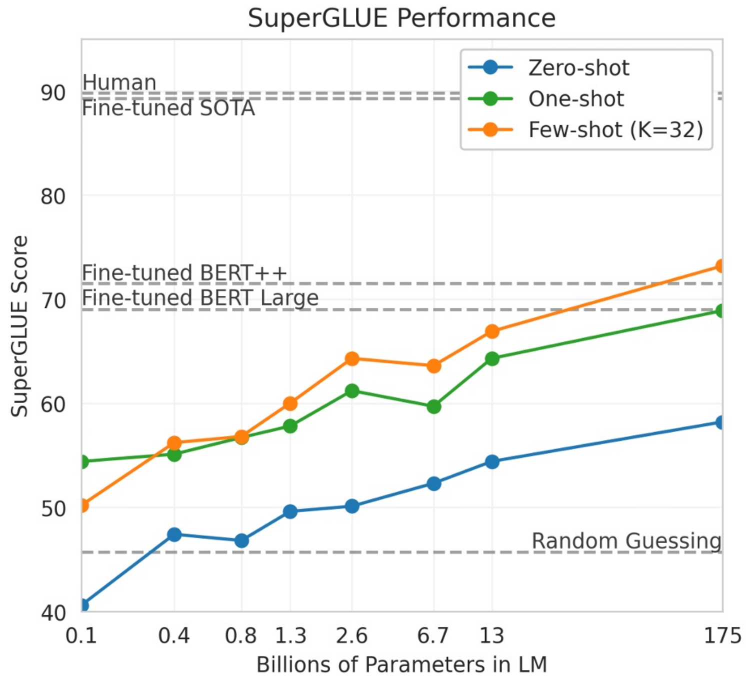
- GPT-2 model size: 1.5 billion parameters
- Pre-trained models can be very, very large (GPT-3 has 175 billion parameters!) and have very strong text generation abilities.

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Performance of Zero-Shot/Few-Shot GPT-3

	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	**89.0**	**91.0**	**96.9**	**93.9**	**94.8**	**92.5**
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0

	WiC Accuracy	WSC Accuracy	MultiRC Accuracy	MultiRC F1a	ReCoRD Accuracy	ReCoRD F1
Fine-tuned SOTA	**76.1**	**93.8**	**62.3**	**88.2**	**92.5**	**93.3**
Fine-tuned BERT-Large	69.6	64.6	24.1	70.0	71.3	72.0
GPT-3 Few-Shot	49.4	80.1	30.5	75.4	90.2	91.1



Next Class: Scaling Up Language Models