Transformers & Pretraining

CS 5539: Advanced Topics in Natural Language Processing

https://shocheen.github.io/courses/advanced-nlp-fall-2024

The Ohio State University

Slide Credits: Daniel Kashabi, Arman Cohen

Logistics

- Foundation Homework – how did everyone do?

- Course Project: Have you formed teams?
  - Email me team names by tonight

- Compute Resources for projects
  - OSC compute (should be assigned this week)
  - Can also use Google Collab / Google Cloud (free student accounts)
  - API access: Azure AI (free student account)

- No office hours this week – Please email to schedule next week

Paper presentations

- The assignments for the next two weeks are up:
  - [CS 5539: List of papers / assignments - Google Sheets]
  - 8 people present each week (2 sessions, 4 roles)

- Please submit your questions / discussion points the night before (Sunday night 11.59 ET)

Recap from last class

- What are language models
    - Distributions over sequences of [words, character, tokens ....]
- What are they useful for
    - Measure likelihood of given sequence, ranking different sequences, generating sequences, and more
- How do you measure if a given language model is good
    - Perplexity
- How do you train a language model
    - N-gram LMs
    - Neural LMs – Recurrent NNs

This Class: Transformer based Language Models

- Transformer Architecture
  - Attention is all you need: encoder-decoder architecture

- Transfer Learning: Pretraining / Finetuning paradigm
  - Main Paper: BERT (Encoder only model)
  - Guest Stars: T5 (Encoder/decoder model), GPT2 (decoder only model)

The cat sat on the mat

$$\mathbf{P}(\underbrace{\text{mat}}_{\text{next word}}\,|\,\underbrace{\text{The cat sat on the}}_{\text{context or prefix}})$$

$$\mathbf{P}(\underbrace{X_t}_{\text{next word}} | \underbrace{X_1, \ldots, X_{t-1}}_{\text{context}})$$

$$\underbrace{\mathbf{P}(X_t}_{\text{next word}} | \underbrace{X_1, \ldots, X_{t-1}}_{\text{context}})$$

But more broadly,

$$\mathrm{P}(X_1, \ldots, X_N)$$

$$= \underbrace{\prod_t P(X_\mathrm{t} | X_1, \ldots, X_{t-1})}_{\text{Chain rule}}$$

$$\mathbf{P}(\underbrace{X_t}_{\text{next word}} | \underbrace{X_1, \dots, X_{t-1}}_{\text{context}})$$

But more broadly,

$$\mathrm{P}(X_1, \dots, X_N)$$

A variant

$$\underbrace{\mathrm{P}(X_1, \dots, X_\mathrm{N} \,|\, \underbrace{Y_1, \dots, Y_M)}_{\text{additional input}}}_{\text{Conditional Language Model}}$$

Language Models: N-grams

- Probabilistic n-gram models of text generation [Jelinek+ 1980's, …]
  - Applications: Speech Recognition, Machine Translation

LMs w/ Recurrent Neural Nets

- Core idea: apply a model repeatedly

outputs — output distribution

$$\hat{\boldsymbol{y}}^{(t)} = \mathrm{softmax}\left(\boldsymbol{U}\boldsymbol{h}^{(t)} + \boldsymbol{b}_2\right) \in \mathbb{R}^{|V|}$$

hidden states

$$\boldsymbol{h}^{(t)} = \sigma\left(\boldsymbol{W}_h\boldsymbol{h}^{(t-1)} + \boldsymbol{W}_e\boldsymbol{e}^{(t)} + \boldsymbol{b}_1\right)$$

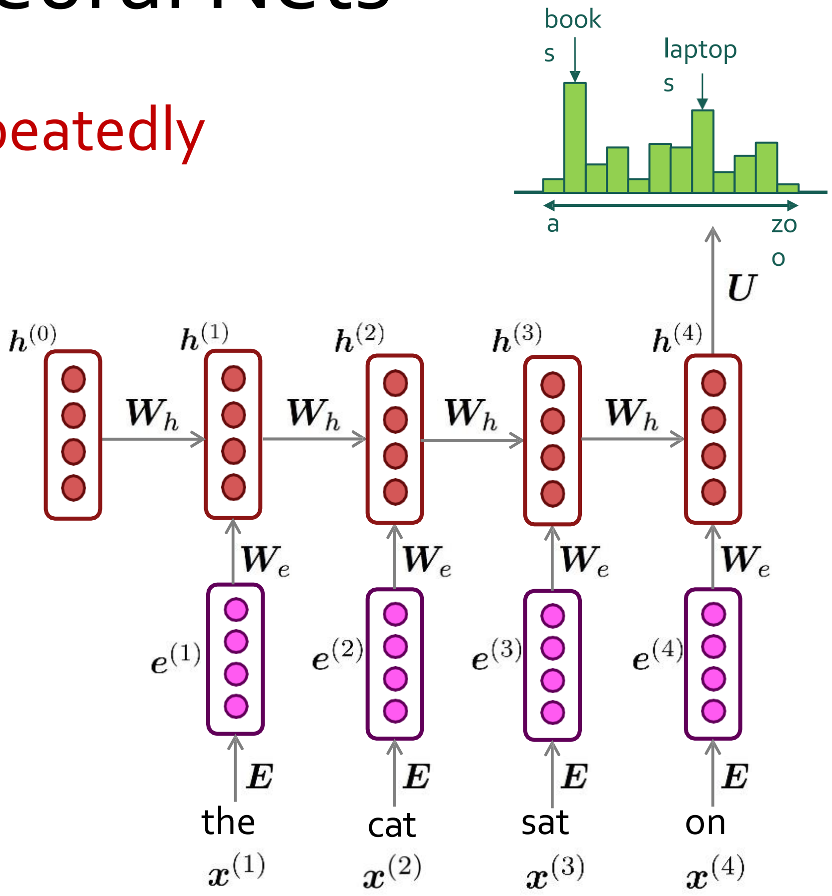
$\boldsymbol{h}^{(0)}$ is the initial hidden state

Input embedding — word embeddings

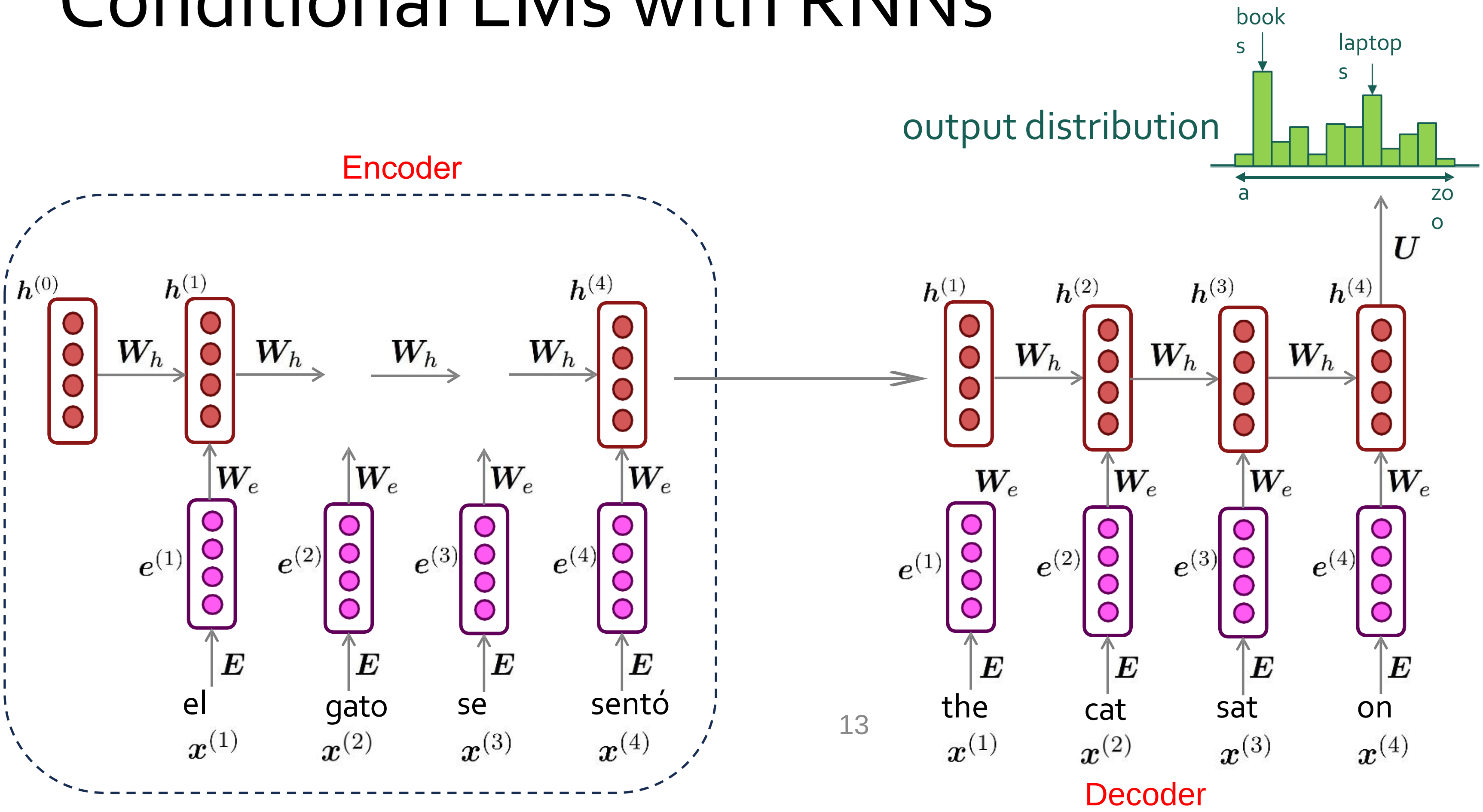
$$\boldsymbol{e}^{(t)} = \boldsymbol{E}\boldsymbol{x}^{(t)}$$

words / one-hot vectors

$$\boldsymbol{x}^{(t)} \in \mathbb{R}^{|V|}$$



Conditional LMs with RNNs

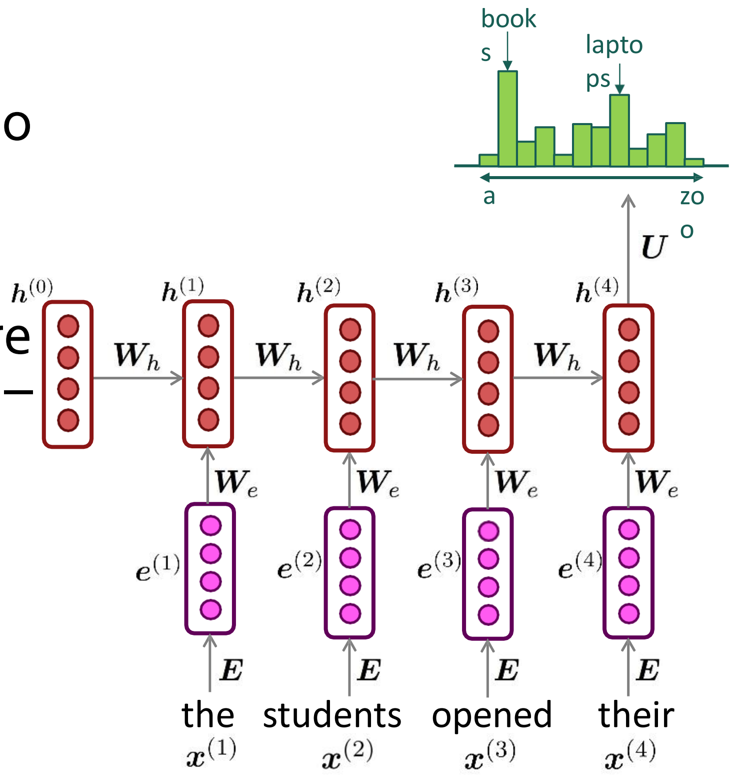


13
13

RNNs: Cons

- Recurrent computation is slow, difficult to parallelize.

- Each state is expected to store the entire information from the previous context – poor performance



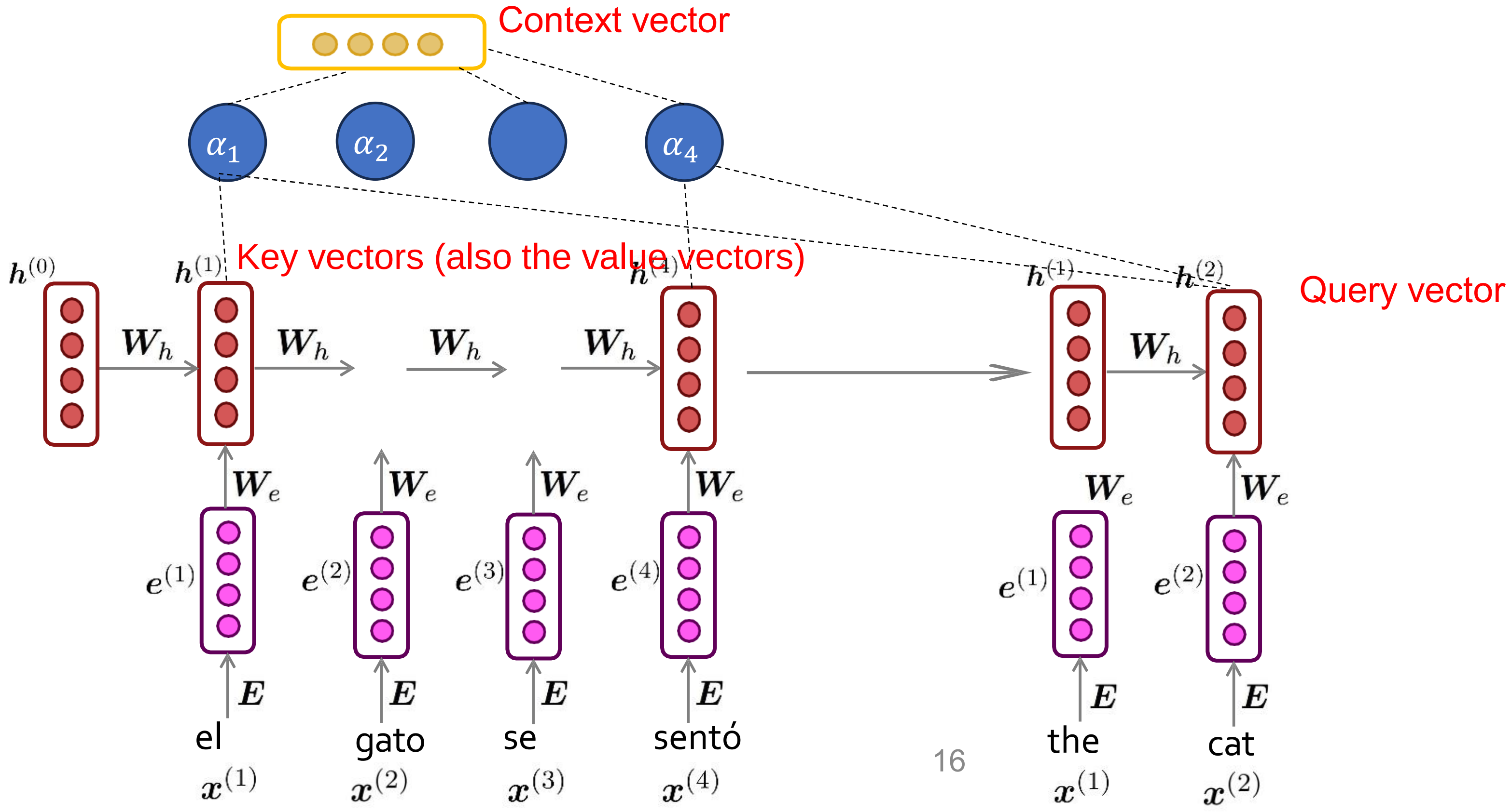
[adopted from Chris Manning]

RNNs

- What if the decoder at each step pays “attention” to a distribution of all of encoder’s hidden states?

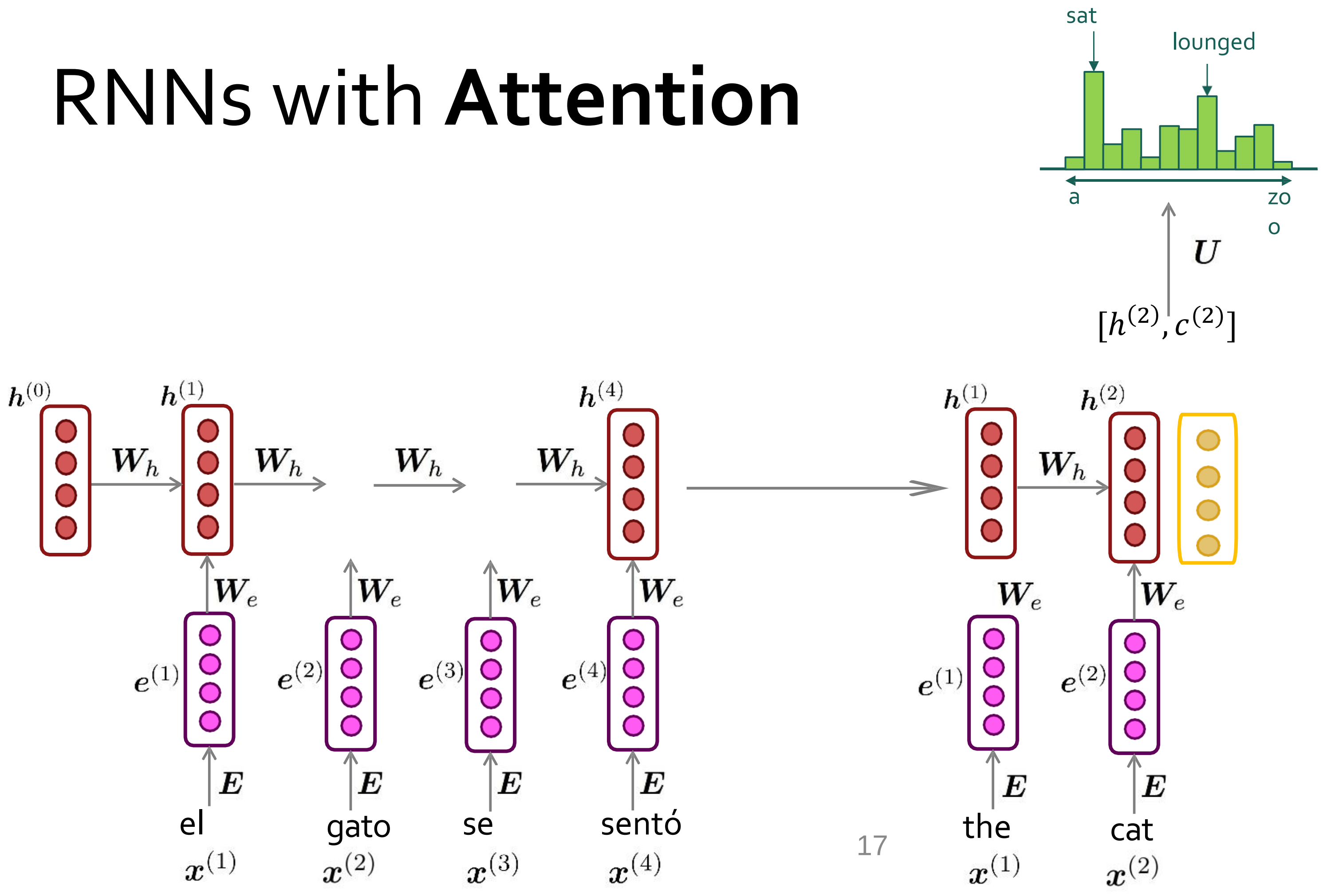
- Intuition: when we (humans) translate a sentence, we don’t just consume the original sentence then regurgitate in a new language; we continuously look back at the original while focusing on different parts

RNNs with **Attention**



16

RNNs with **Attention**



17

RNNs with Attention

- Attention allowed modelling longer context and obtain higher performance

- But
    - It is still slow because of linear computation in RNN
    - It still has gradient vanishing/exploding issues

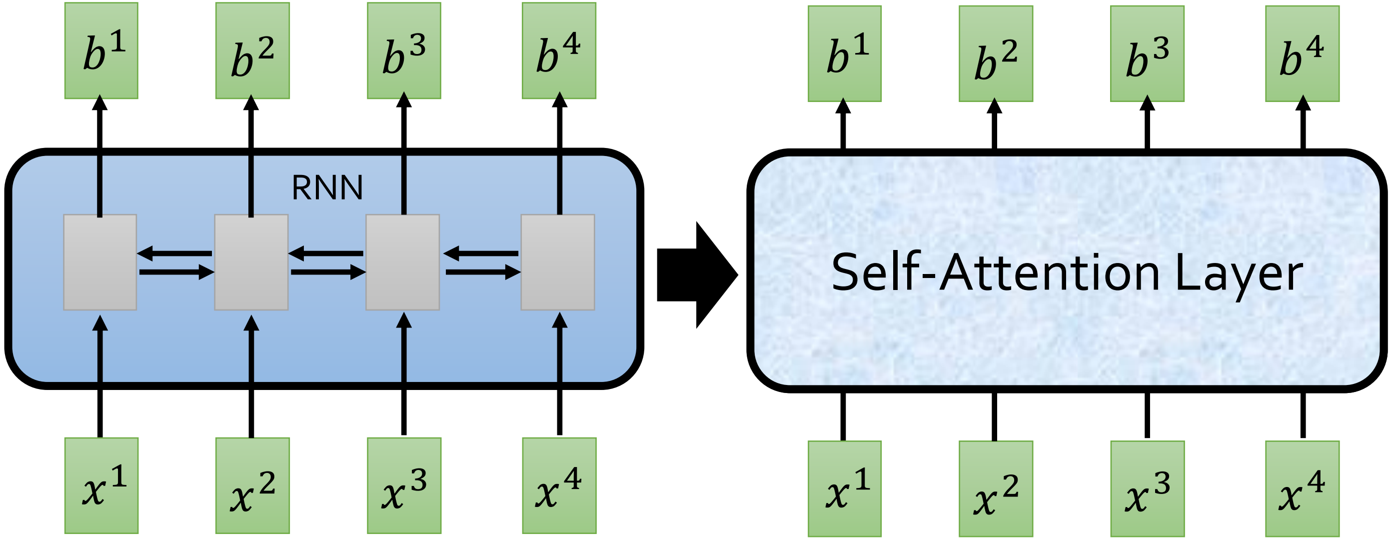
- Solution: what if we removed the RNN component and only use attention
    - Attention is all you need (Vaswani et al 2017)

Transformers

- Replace the linear part with **self-attention**
- Introduce **residual connections** to improve gradient flow
- Introduce **positional embeddings** to encode sequential order

Self-Attention

- b^t is obtained based on the whole input sequence.
- can be parallelly computed.



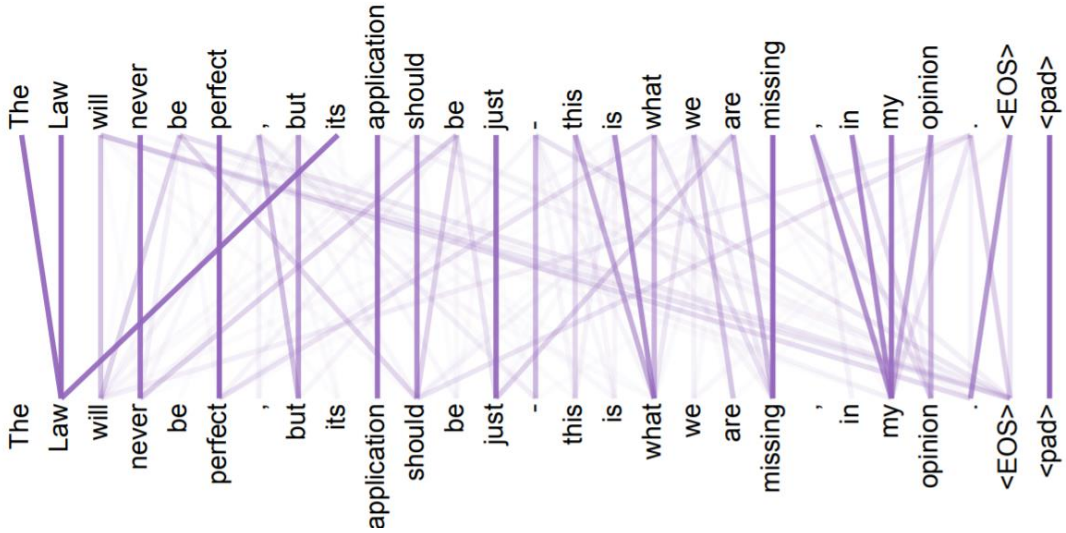
Idea: replace any thing done by RNN with self-attention.

"Neural machine translation by jointly learning to align and translate" Bahdanau etl. 2014;
"Attention is All You Need" Vaswani et al. 2017

[adopted from Hung-yi Lee]

Attention

- <u>Core idea</u>: on each step, *use direct connection* to *focus ("attend") on a particular part* of the context.



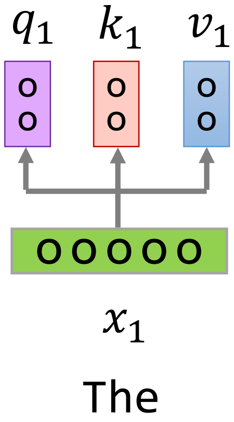
[Vaswani et al. 2017: https://arxiv.org/abs/1706.03762]

Defining Self-Attention

- **Terminology:**
  - **Query**: to match others
  - **Key**: to be matched
  - **Value**: information to be extracted

- **Definition:** Given a set of vector **values**, and a vector **query**, *attention* is a technique to compute a weighted sum of the **value**, dependent on the **query**.

[Vaswani et al. 2017: https://arxiv.org/abs/1706.03762]



q: query (to match others)

$$q_t = W^q x_t$$

k: key (to be matched)

$$k_t = W^k x_t$$

v: value (information to be extracted)

$$v_t = W^v x_t$$

q: query (to match others)

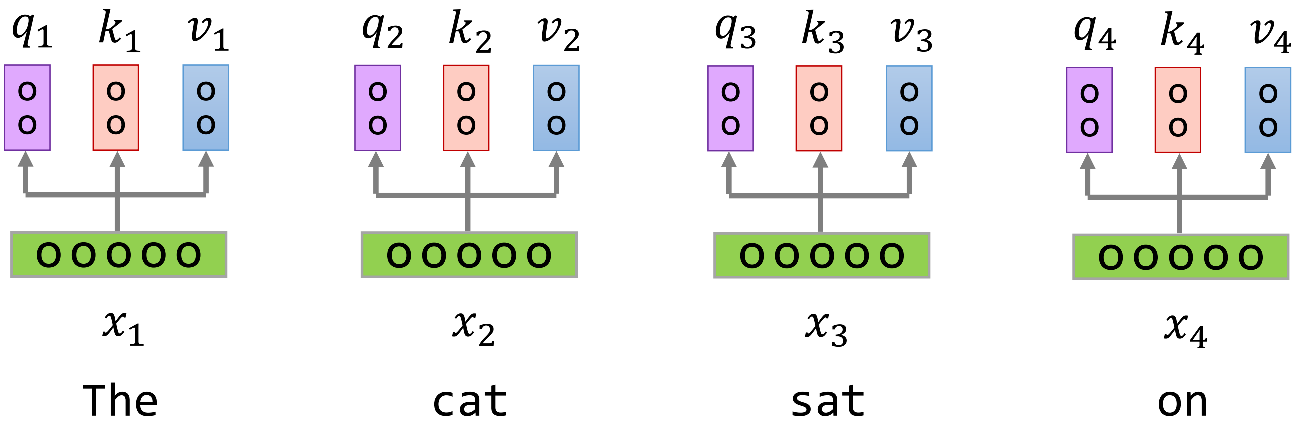
$$q_t = W^q x_t$$

k: key (to be matched)

$$k_t = W^k x_t$$

v: value (information to be extracted)

$$v_t = W^v x_t$$



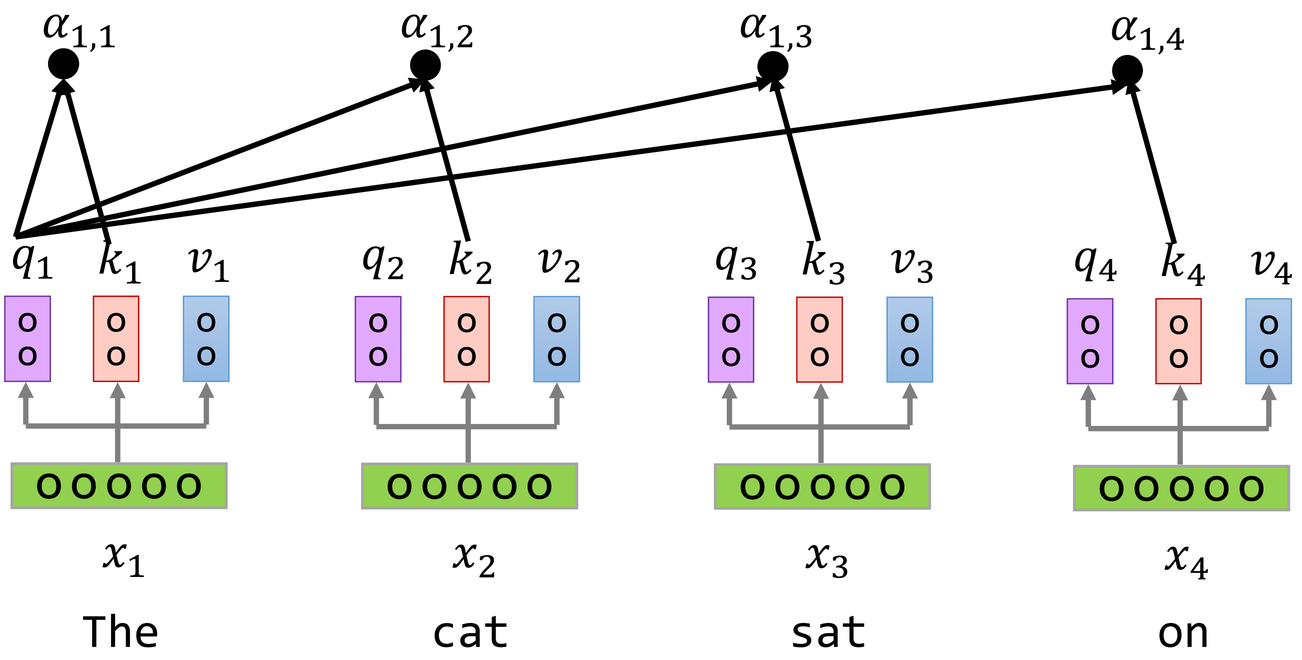
$$\alpha_{1,t} = \underbrace{q^1 \cdot k^t / \alpha}_{\text{Scaled dot product}}$$

q: query (to match others)

k: key (to be matched)

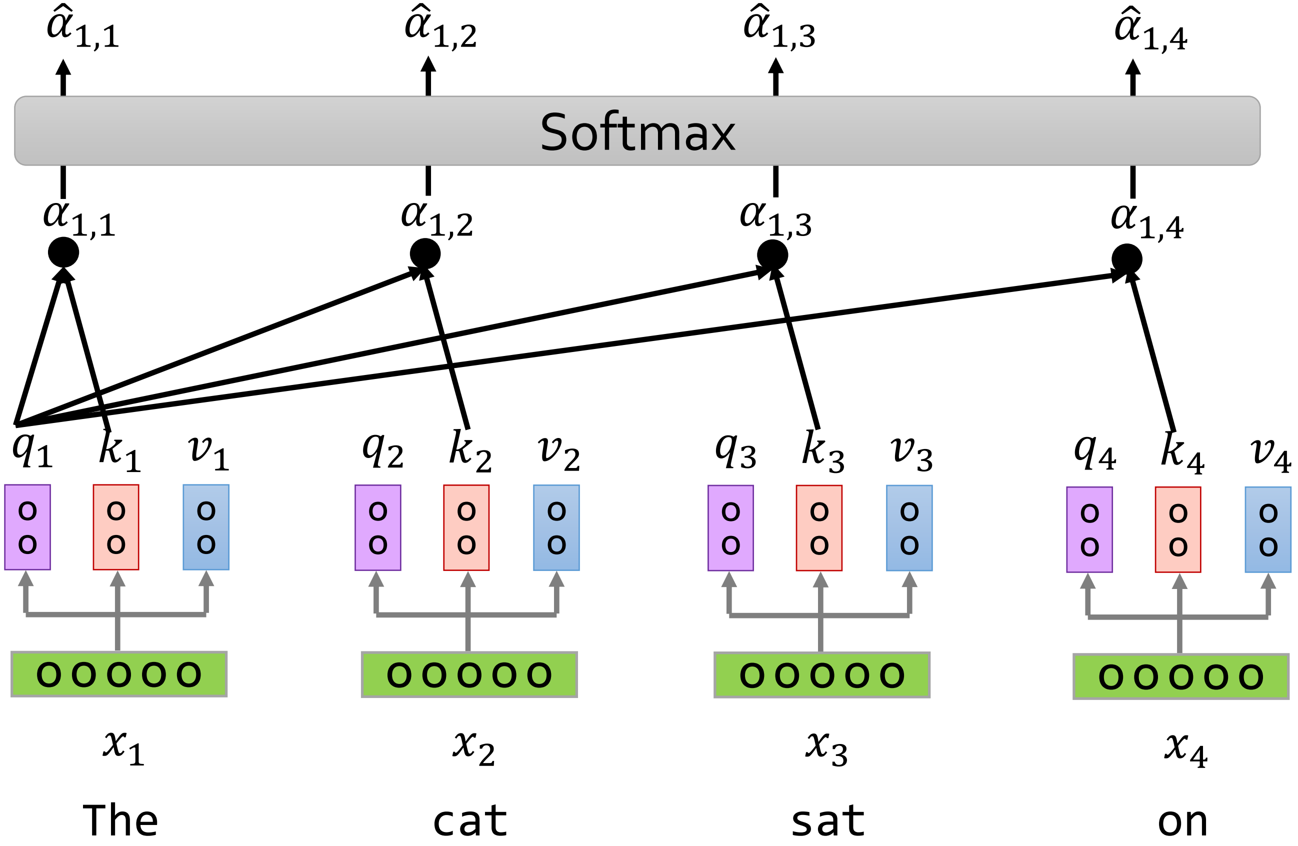
v: value (information to be extracted)

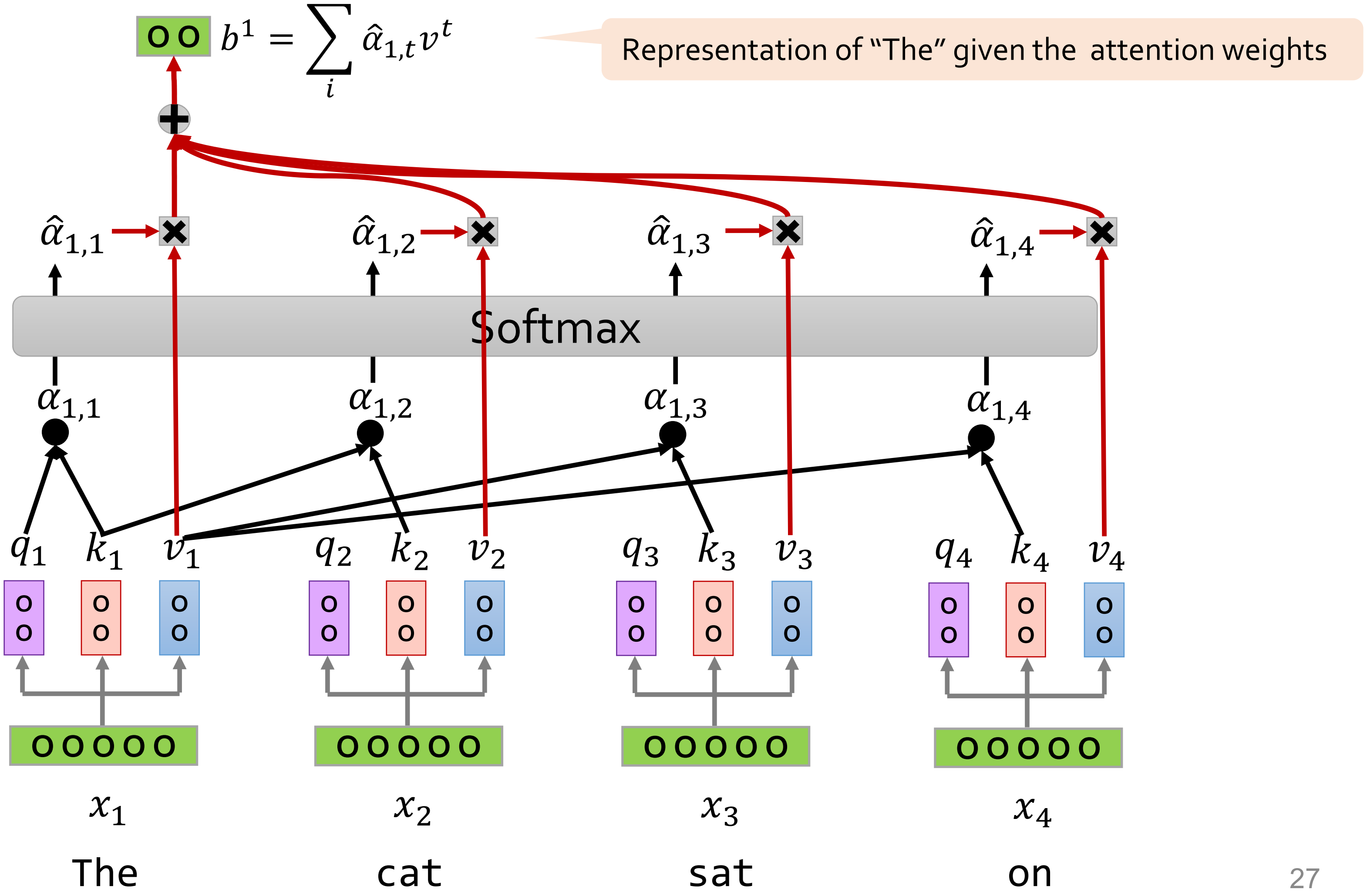
How much should "The" attend to other positions?



$$\sigma(z)_t = \frac{exp(z_t)}{\sum_j exp(z_j)}$$

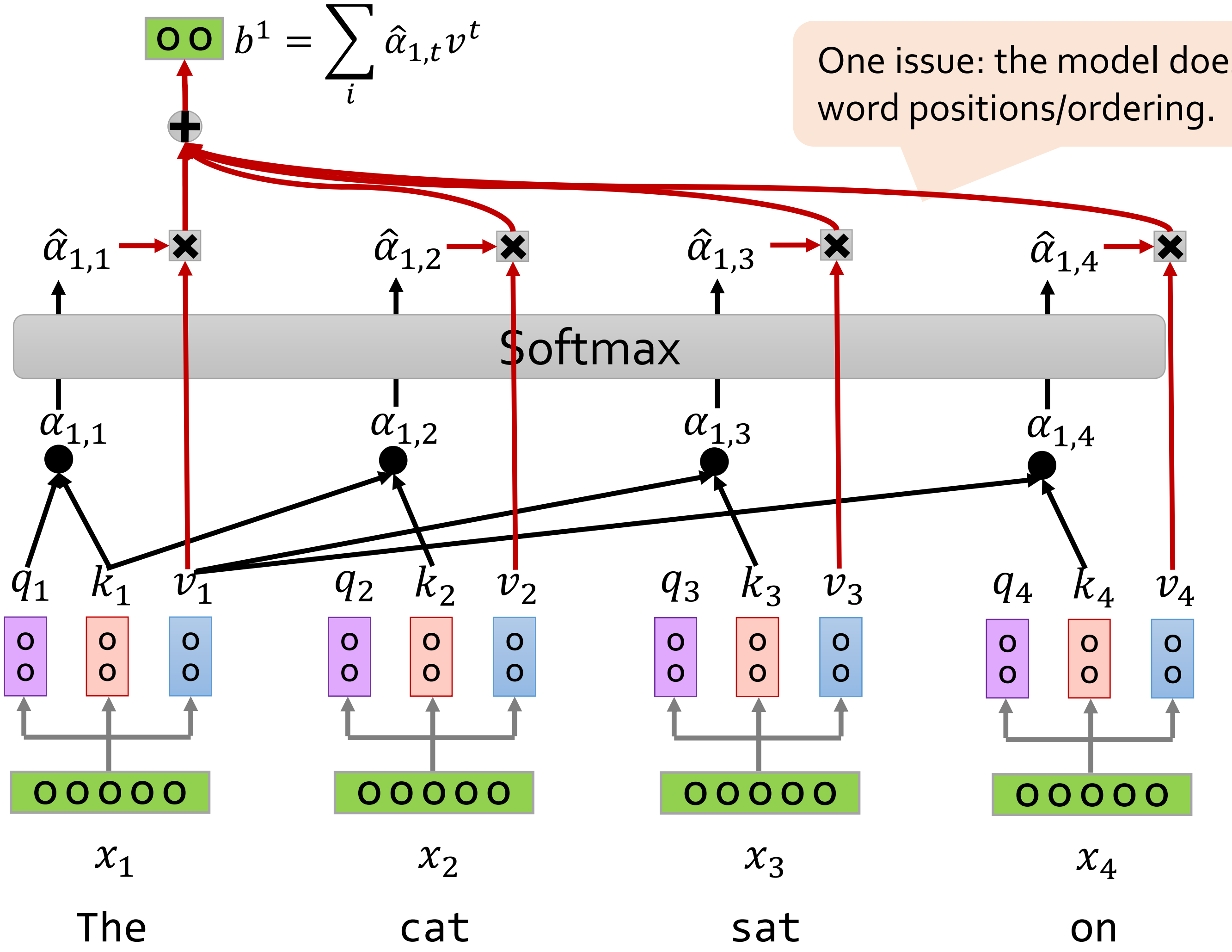
How much should "The" attend to other positions?





$b^1 = \sum_i \hat{\alpha}_{1,t} v^t$
Representation of "The" given the attention weights
$\hat{\alpha}_{1,1}$
$\hat{\alpha}_{1,2}$
$\hat{\alpha}_{1,3}$
$\hat{\alpha}_{1,4}$
Softmax
$\alpha_{1,1}$
$\alpha_{1,2}$
$\alpha_{1,3}$
$\alpha_{1,4}$
q_1 k_1 v_1
q_2 k_2 v_2
q_3 k_3 v_3
q_4 k_4 v_4
x_1
x_2
x_3
x_4
The
cat
sat
on

One issue: the model doesn't know word positions/ordering.



How to encode position information?

- Self attention doesn't have a way to know whether an input token comes before or after another
  - Position is important in sequence modeling in NLP

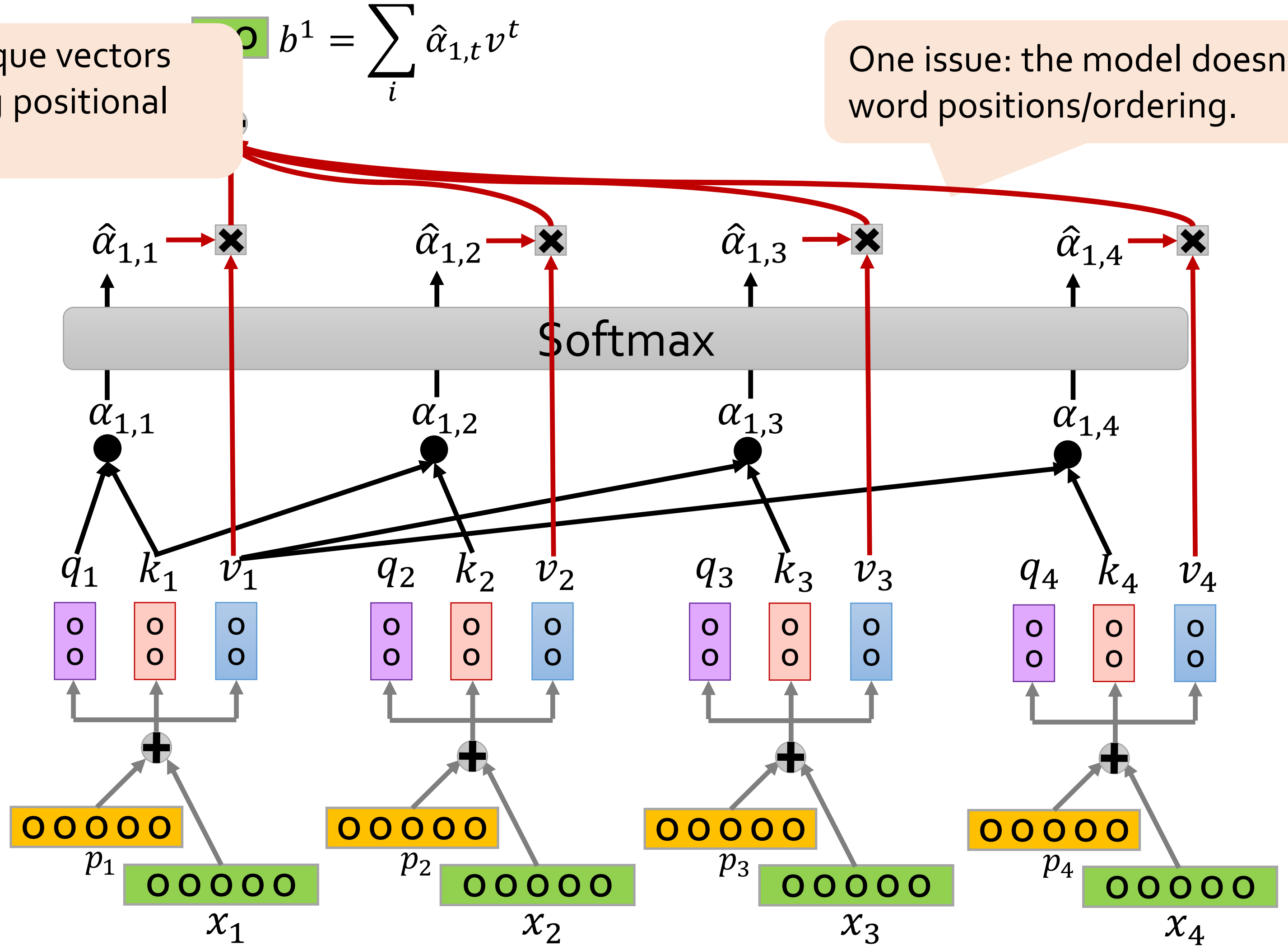
- A way to introduce position information is add individual position encodings to the input for each position in the sequence

$$x_t = x_t + pos_t$$

Where pos_i is a position vector

pos_i are unique vectors representing positional information

One issue: the model doesn't know word positions/ordering.



Properties of a good positional embedding

- It should output a unique encoding for each time-step (word’s position in a sentence)
- Distance between any two time-steps should be consistent across sentences with different lengths.
    - The cat sat on the mat
    - The happy cat sat on the mat
- Our model should generalize to longer sentences without any efforts. Its values should be bounded.
- It must be deterministic.

Absolute position embeddings

- Learned positions embeddings:
  - Maximum length that can be presented is limited
  - Difficult to encode relative positions
    - The cat sat on the mat
    - The happy cat sat on the mat

Functional position embeddings

$$\vec{p_t}^{(i)} = f(t)^{(i)} := \begin{cases} \sin(\omega_k . t), & \text{if } i = 2k \\ \cos(\omega_k . t), & \text{if } i = 2k \end{cases}$$

where

$$\omega_k = \frac{1}{10000^{2k/d}}$$

$$\vec{p_t} = \begin{bmatrix} \sin(\omega_1 . t) \\ \cos(\omega_1 . t) \\ \\ \sin(\omega_2 . t) \\ \cos(\omega_2 . t) \\ \\ \vdots \\ \\ \sin(\omega_{d/2} . t) \\ \cos(\omega_{d/2} . t) \end{bmatrix}_{d \times 1}$$

The frequencies are decreasing along the vector dimension. It forms a geometric progression from 2π to 10000·2π on the wavelengths.

Sinusoidal Embeddings: Intuition

0 :	0	0	0	0	8 :	1	0	0	0
1 :	0	0	0	1	9 :	1	0	0	1
2 :	0	0	1	0	10 :	1	0	1	0
3 :	0	0	1	1	11 :	1	0	1	1
4 :	0	1	0	0	12 :	1	1	0	0
5 :	0	1	0	1	13 :	1	1	0	1
6 :	0	1	1	0	14 :	1	1	1	0
7 :	0	1	1	1	15 :	1	1	1	1

[Transformer Architecture: The Positional Encoding - Amirhossein Kazemnejad's Blog]

Variants of Positional Embeddings

- Rotary Positional Embeddings (RoPE): [2104.09864] RoFormer: Enhanced Transformer with Rotary Position Embedding (arxiv.org)

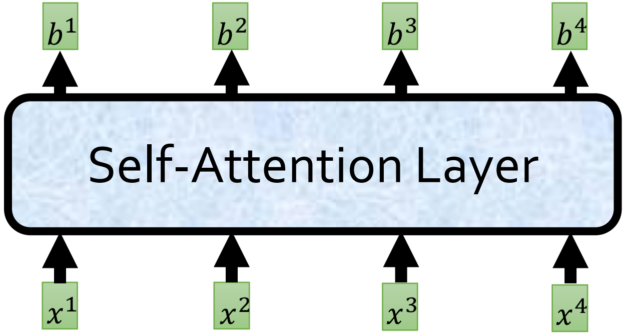
- AliBi: [2108.12409] Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation (arxiv.org)

- No embeddings(!?): [2203.16634] Transformer Language Models without Positional Encodings Still Learn Positional Information (arxiv.org)

Self-Attention: Back to Big Picture

- Attention is a way to focus on particular parts of the input
- Can write it in matrix form:

$$\boldsymbol{b} = \mathrm{softmax}\left(\frac{QK^{\mathrm{T}}}{\alpha}\right)V$$



- Efficient implementations
- Better at maintaining long-distance dependencies in the context.

Self-Attention

$$\boldsymbol{b} = \mathrm{softmax}\left(\frac{QK^{\mathrm{T}}}{\alpha}\right)V$$

hardmaru
@hardmaru

The most important formula in deep learning after 2018

Self-Attention

What is self-attention? Self-attention calculates a weighted average of feature representations with the weight proportional to a similarity score between pairs of representations. Formally, an input sequence of n tokens of dimensions d, $X \in \mathbf{R}^{n \times d}$, is projected using three matrices $W_Q \in \mathbf{R}^{d \times d_q}$, $W_K \in \mathbf{R}^{d \times d_k}$, and $W_V \in \mathbf{R}^{d \times d_v}$ to extract feature representations Q, K, and V, referred to as query, key, and value respectively with $d_k = d_q$. The outputs Q, K, V are computed as

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V. \tag{1}$$

So, self-attention can be written as,

$$S = D(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_q}}\right)V, \tag{2}$$

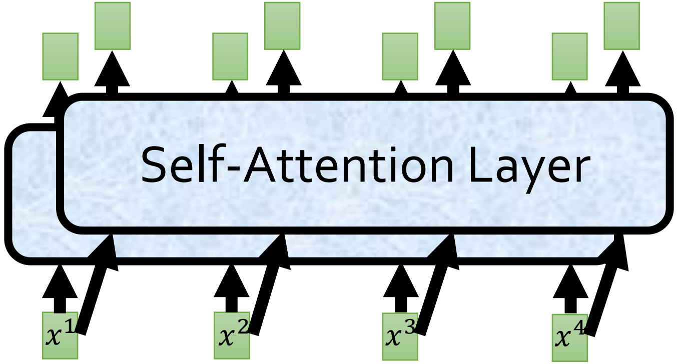
where softmax denotes a *row-wise* softmax normalization function. Thus, each element in S depends on all other elements in the same row.

9:08 PM · Feb 9, 2021 · Twitter Web App

553 Retweets **42** Quote Tweets **3,338** Likes

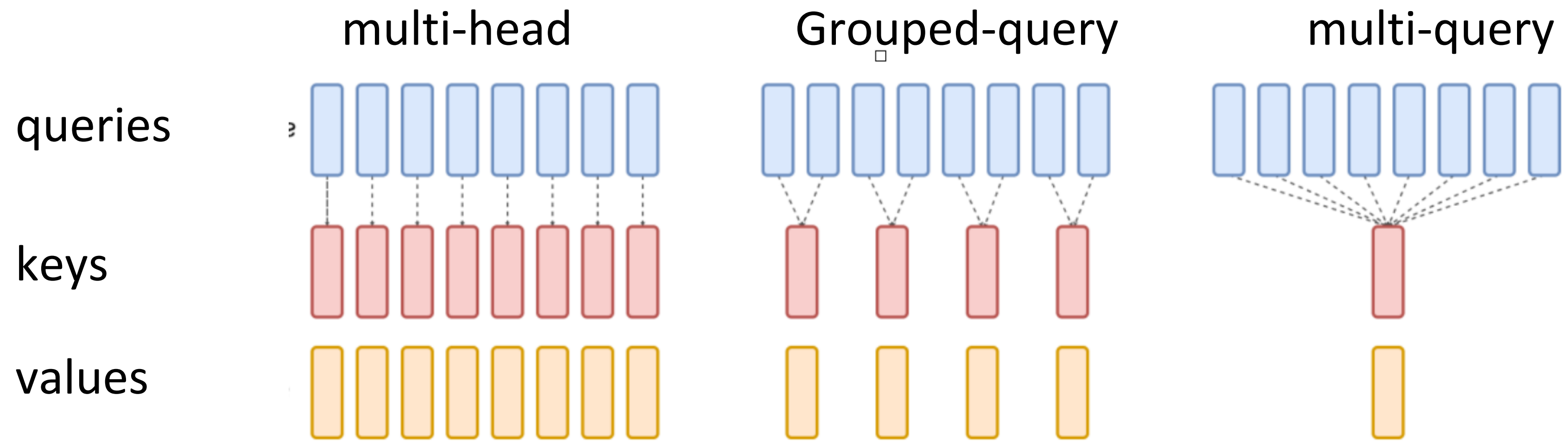
Multi-Headed Self-Attention

- Multiple parallel attention layers is quite common.
  - Each attention layer has its own parameters.



[Vaswani et al. 2017]

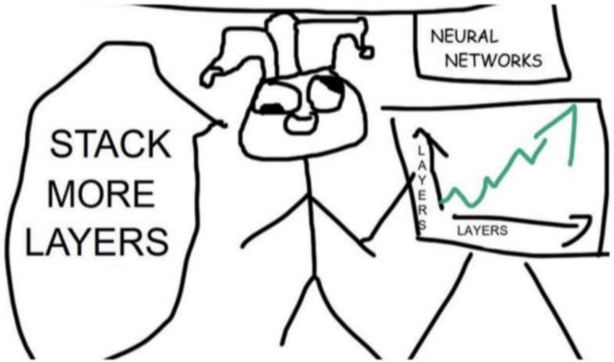
Variants of attention

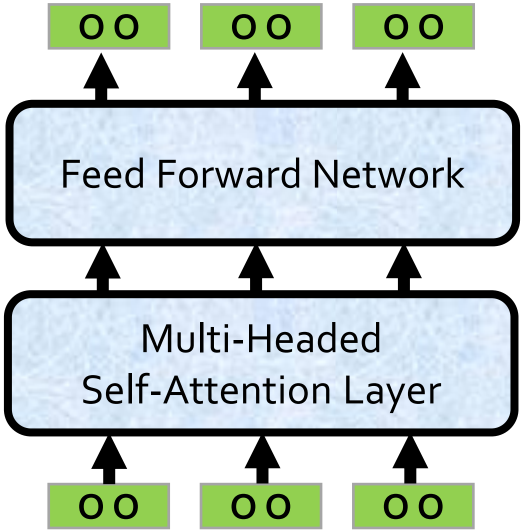


GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints (Ainslie et al., 2023)

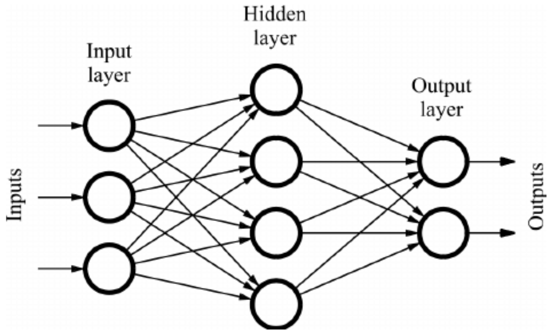
How Do We Make it Deep?

- Add a feed-forward network on top it to add more capacity/expressivity.
- Repeat!





Feedforward Net: Refresher



A fully-connected network of nodes and weights.

Feed forward layer

- A position-wise transformation consisting of:
  - A linear transformation, non-linear activation f (e.g., ReLU), and another linear transformation.

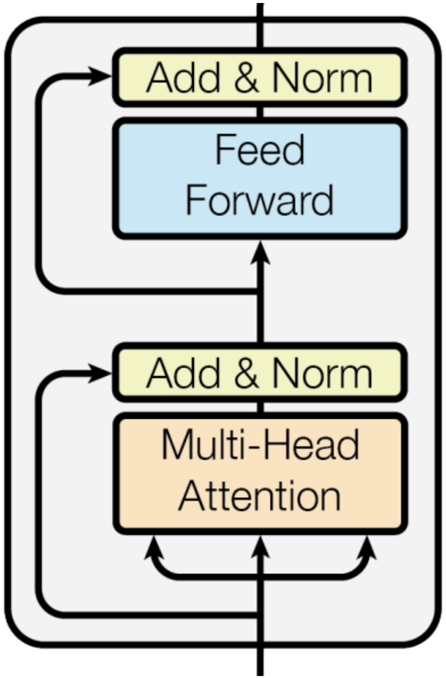
$$FF(c) = f(cW_1 + b_1)W_2 + b_2$$

- This allows the model to apply another transformation to the contextual representations (or “post-process” them)

- Usually the dimensionality of the hidden feedforward layer is 2-8 times larger than the input dimension

A transformer block

out



x: input sequence

$$\text{out} = LayerNorm(c' + \text{FF}(c'))$$

$$FF(c') = f(c'W_1 + b_1)W_2 + b_2$$

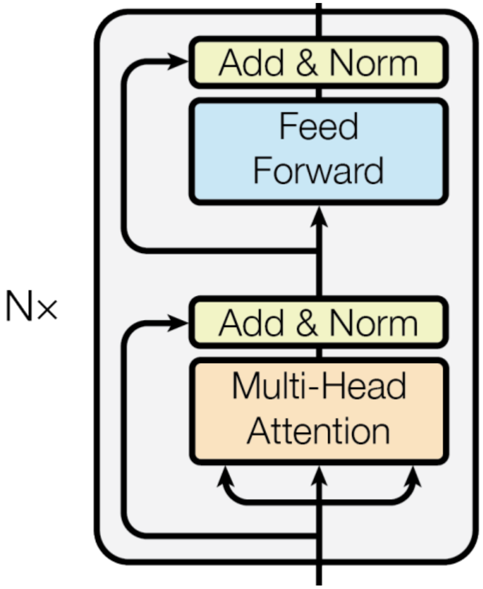
$$c' = LayerNorm(c + x)$$

$$c = \text{MultiHeadAttention}(\text{q}, \text{k}, \text{v})$$

$$q, k, v = \text{QKV\_Projection}(x)$$

Transformer stack

- A stack of N transformer blocks (organized in N layers)

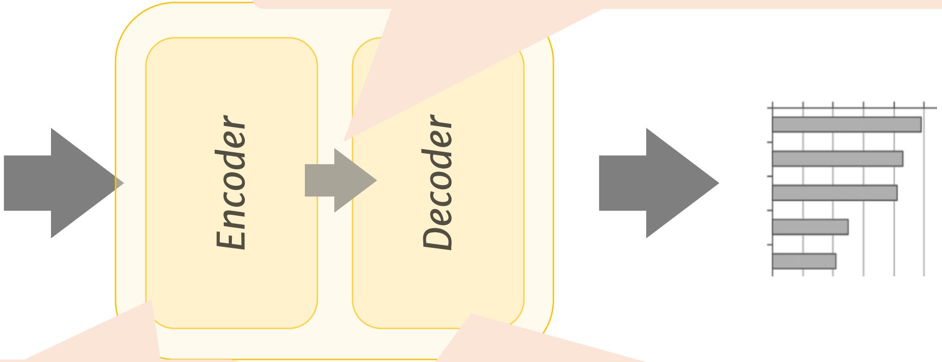


Encoder-Decoder Architectures

- Original transformer had two sub-models.

Representation (compression) of the context

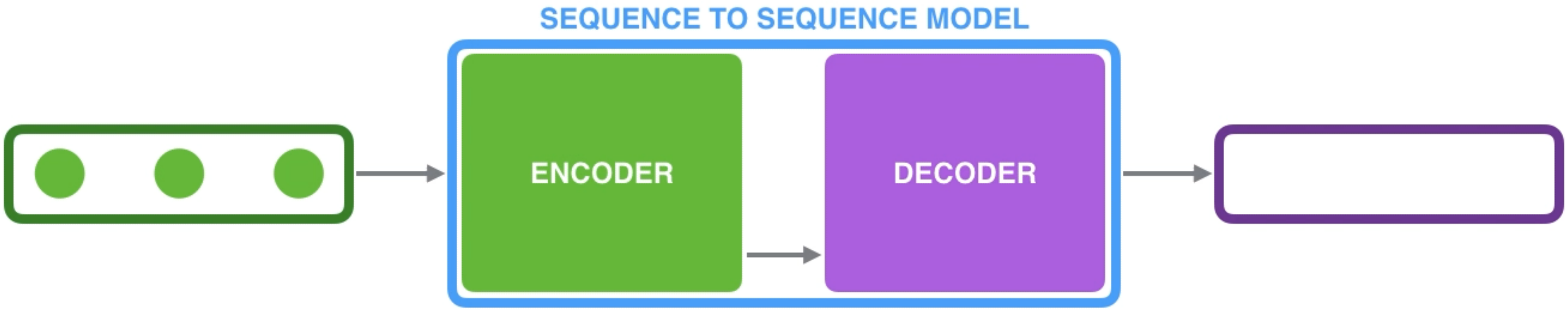
`El gato se sento`



Processes the context and compiles it into a vector.

Produces the output sequence item by item using the representation of the context.

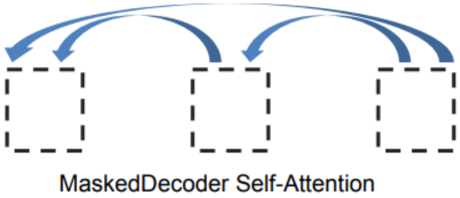
Encoder-Decoder Architectures

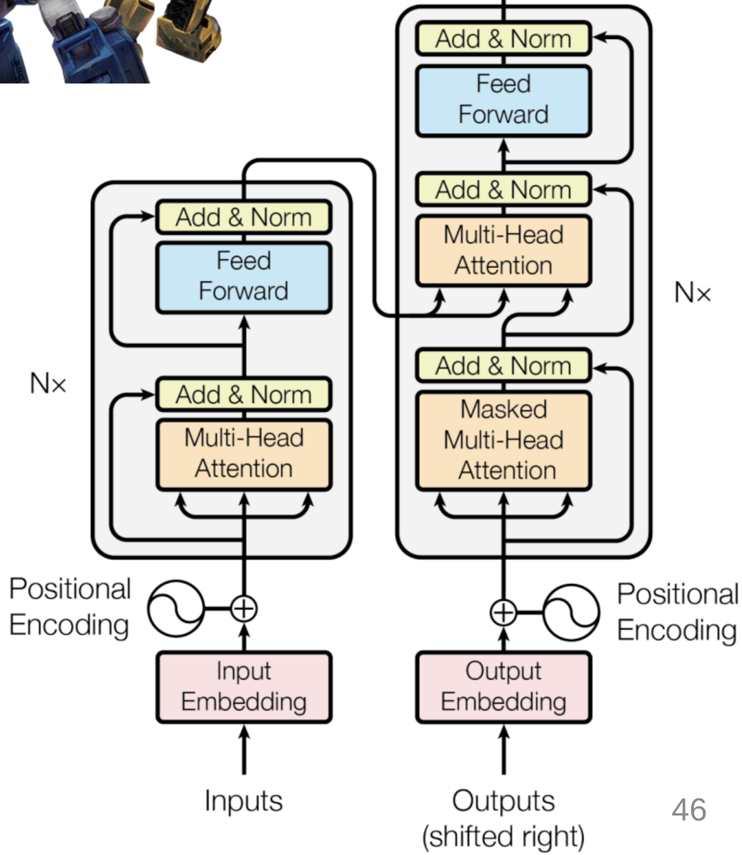


https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

Transformer [Vaswani et al. 2017]

- An encoder-decoder architecture built with attention modules.
- 3 forms of attention





Transformers as machine translation models

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Impact of Transformers

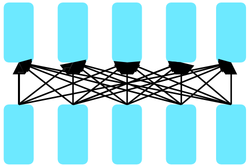
- Let to better predictive models of language ala GPTs!

Model	Layers	Heads	Perplexity
LSTMs (Grave et al., 2016)	-	-	40.8
QRNNs (Merity et al., 2018)	-	-	33.0
Transformer	16	16	19.8

["Efficient Content-Based Sparse Attention with Routing Transformers" Roy et al. 2020]

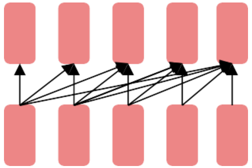
Impact of Transformers

- A building block for a variety of LMs



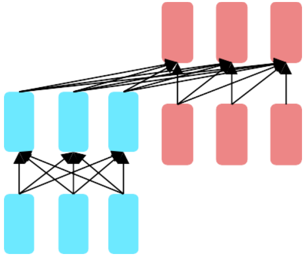
Encoders

- ❖ **Examples:** BERT, RoBERTa, SciBERT.
- ❖ Captures bidirectional context. How do we pretrain them?



Decoders

- ❖ **Examples:** GPT-2, GPT-3, Llama models, and many many more
- ❖ Other name: **causal or auto-regressive language model**
- ❖ Nice to generate from; can't condition on future words



Encoder-Decoders

- ❖ **Examples:** Transformer, T5, BART
- ❖ What's the best way to pretrain them?

Transformer LMs + Scale = LLMs

- 2 main dimensions:
- Model size, pretraining data size

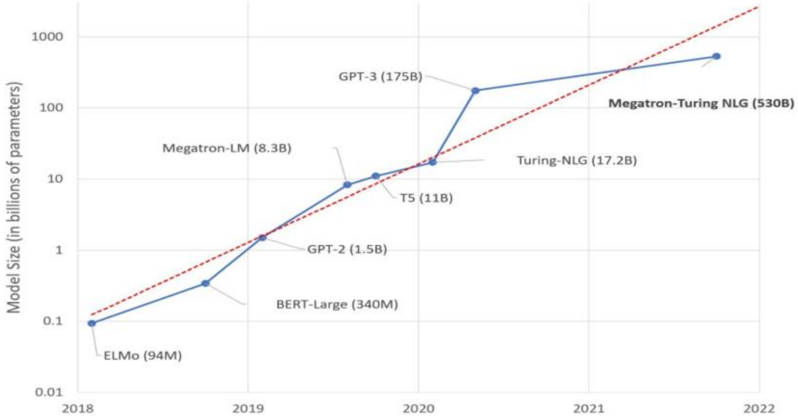
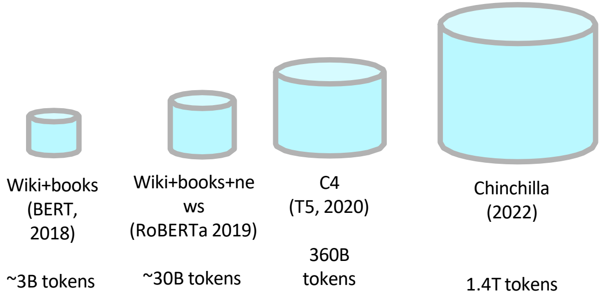
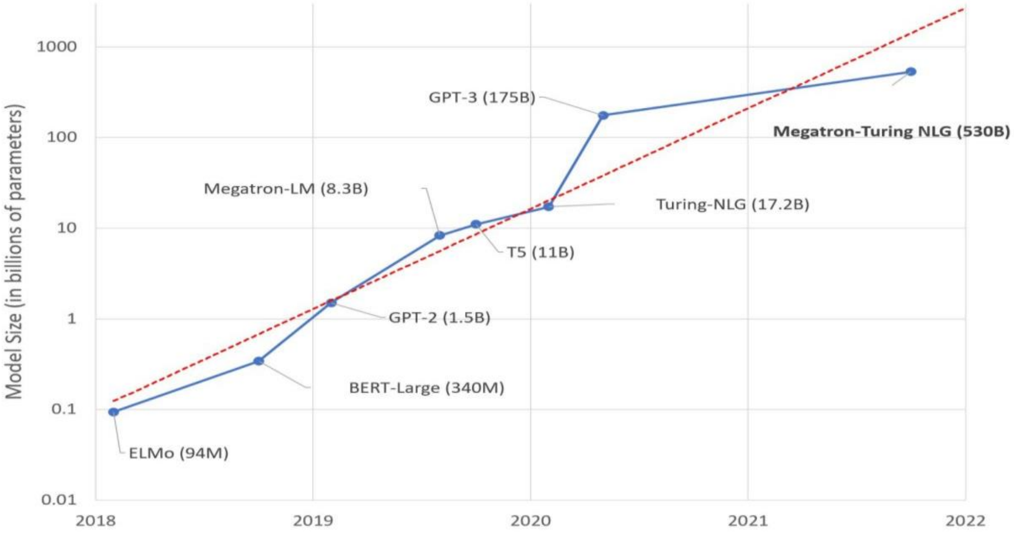


Photo credit: https://www.microsoft.com/en-us/research/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/



Large Language Models

- Not only they improved performance on many NLP tasks, but exhibited new capabilities



Transformers - Summary

- Self-attention + positional embedding + others = NLP go brr

- Much faster to train than any previous architectures, much easier to scale

- Perform on par or better than previous RNN based models
    - Ease of scaling allows to extract much better performance

Questions

Pretraining / Finetuning

BERT / GPT2 / T5

BERT: Motivation

- A typical recipe for any NLP task such as text classification, translation, summarization, parsing etc.
  - Collect training examples (input, output)
  - Train a machine learning model (such as RNN/Transformer based model)

- Most NLP tasks share underlying features
  - Intuitively, all of them involve some level of "understanding"

- Instead of individual models for each task from scratch, can we learn shared representations that can help each task

Motivation

- Build a **language representation system** that can be used to solve different NLP tasks.

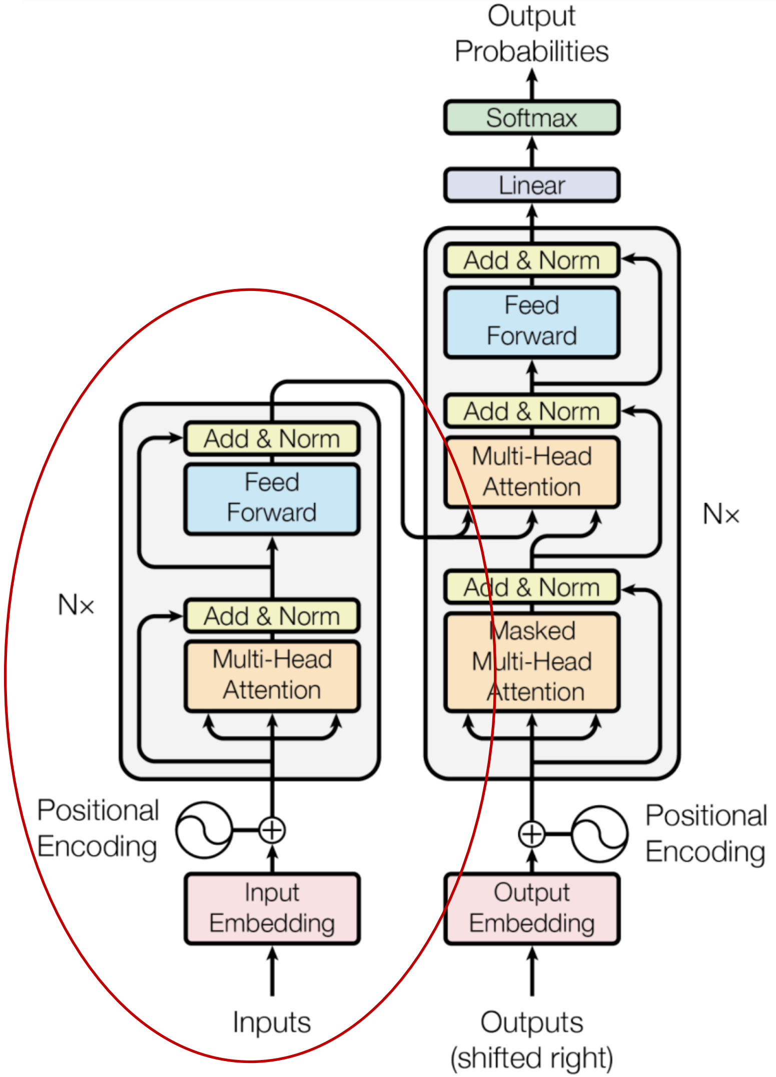
- How to build: **pretraining** on an unlabeled corpus

- How to solve: **finetuning** on a task-specific labeled dataset

BERT

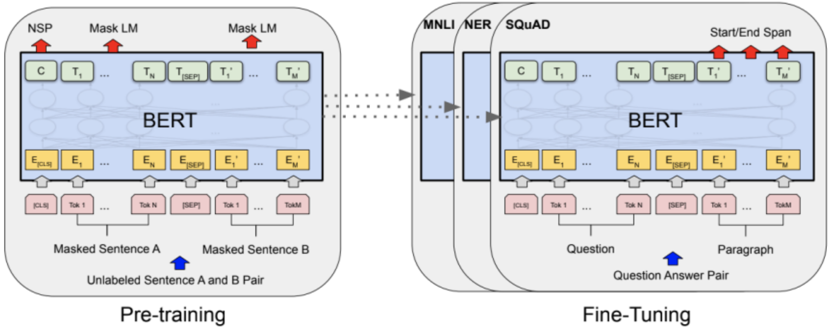
Bidirectional Encoder Representation from Transformer (BERT):

- A stack of multiple transformer encoders
- BERT is a **fast bidirectional** model trained to understand "context"



:Sachin Kumar

Method



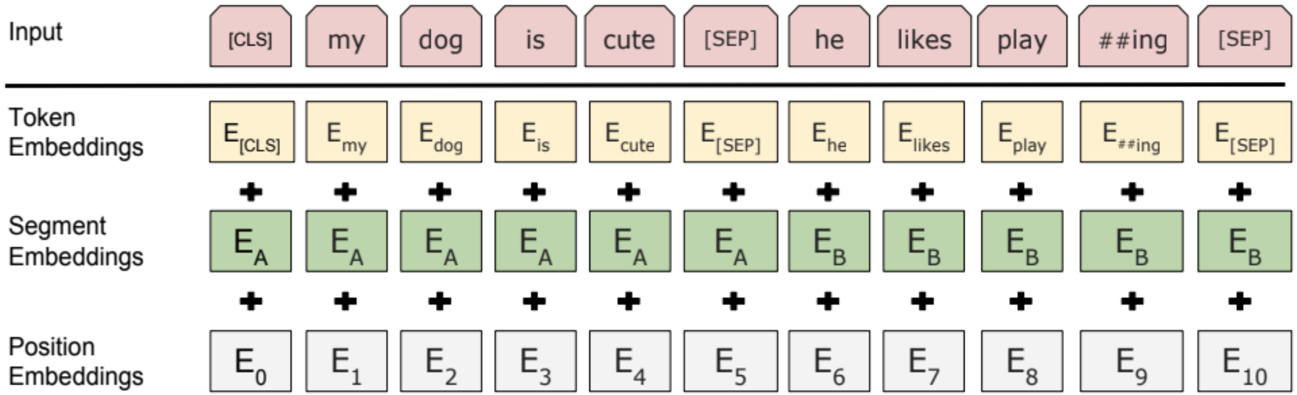
Overview of two steps of training BERT:

- Pre-training:
  - Goal: **Understanding** features in representation space
  - Trains model on unlabeled data over different pre-training tasks (**self-supervised learning**)
- Fine-tuning:
  - Goal: Make pre-trained model **usable** in **downstream tasks**
  - Initialized with pre-trained model parameters
  - Fine-tuned model parameters using labeled data from downstream tasks

:Sachin Kumar

Devlin, 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Method



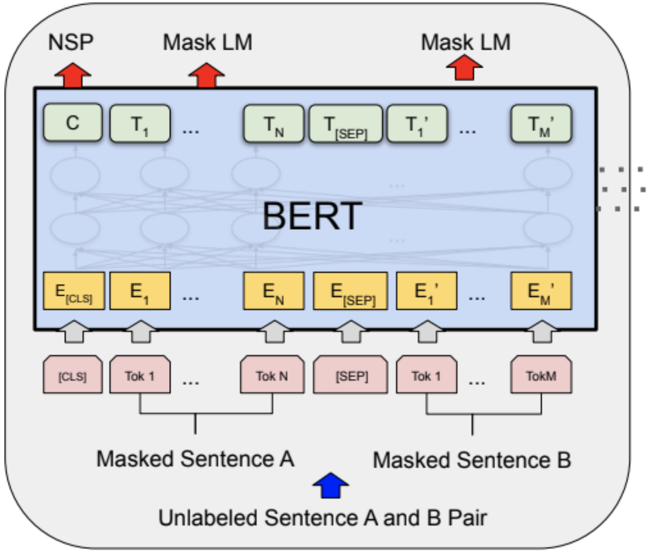
Input:

- **Token**: pre-trained token vocabs ("WordPieces": 30K vocabs/tokens)
    - [CLS]: token beginning sentence, [SEP]: token ending sentence
    - **Segment**: sentence number encoder to vectors
- **Position**: position of words within that sentence
- => Preserve **ordering** sentence inputs for BERT => Robust across downstream tasks

Method

Pre-training BERT:

- Task #1: **Masked Language Model**
    - Inputs: The [Mask1] State University is located in [Mask2] city (E)
    - Outputs: [Mask1] = Ohio, [Mask2] = Columbus (C, T)
    - => Helps understand bi-directional context
- Task #2: **Next Sentence Prediction**
    - Inputs:
        - A: Ohio State is a university (E)
        - B: It is located in Columbus (E)
    - Outputs:
        - Yes: Sentence B follows sentence A (C = 1)
    - => Help understand context across different sentences
- Jointly training as a multi-task classification model



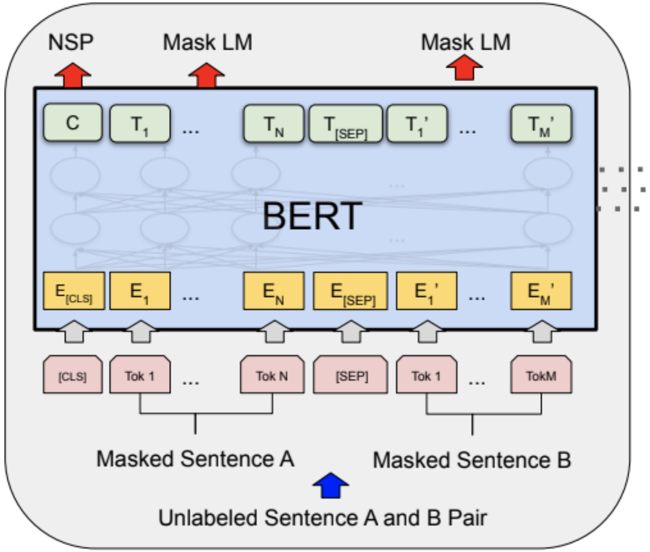
Pre-training

✍: Sachin Kumar

Method

Pre-training BERT: Dataset

Bookscorpus + English Wikipedia (3.3B words)

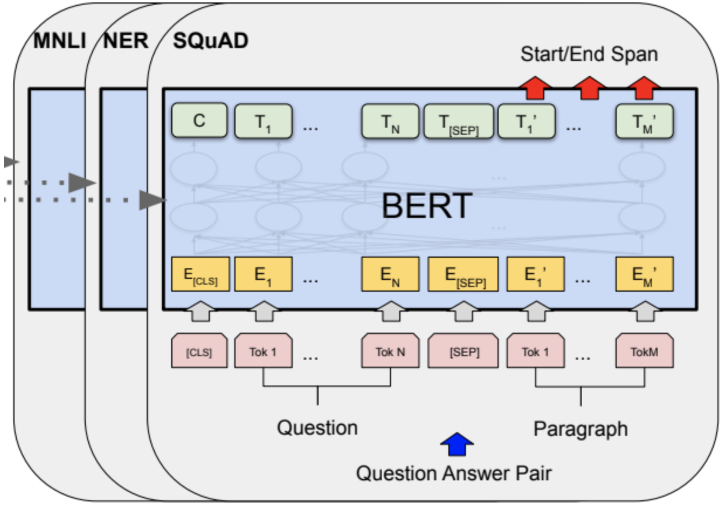


Pre-training

Method

Fine-tuning BERT:

- Replace final layer with a task specific linear layer (classification head)
- Reformat different tasks as sequence or token level classification tasks
- Example in Questions Answering:
  - Inputs: Question, Paragraph
  - Outputs: start and end words that encapsulate the answer



Fine-Tuning

Experiments

Experimental Settings:

- Models:
  - **BERT_base** (#transformer blocks L = 12, #hidden size H = 768, #self-attention heads A = 12): 110M params
  - **BERT_large** (L =24, H = 1024, A = 16): 340M params
- Fine-tuning on 11 NLP tasks over GLUE, SQuAD v1.1, SQuAD v2.0, SWAG dataset

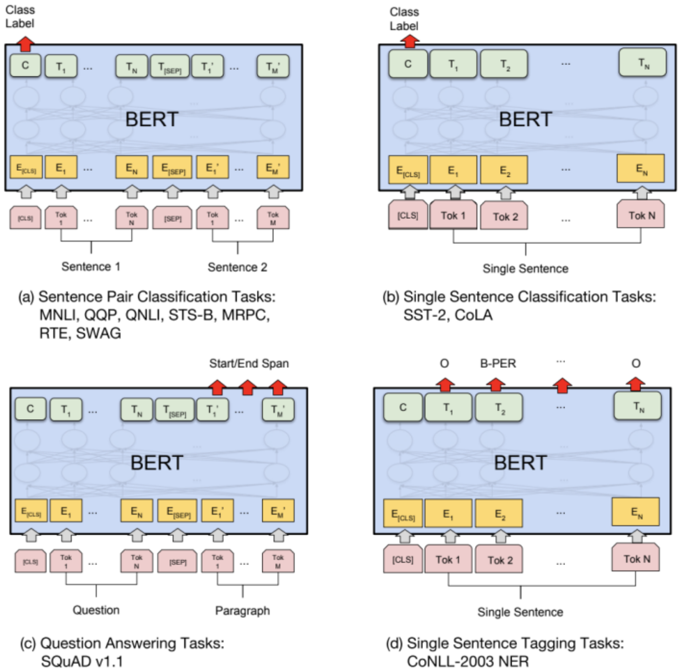


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

Results

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**

Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard).

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$BERT_{BASE}$ (Single)	80.8	88.5	-	-
$BERT_{LARGE}$ (Single)	84.1	90.9	-	-
$BERT_{LARGE}$ (Ensemble)	85.8	91.8	-	-
$BERT_{LARGE}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$BERT_{LARGE}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

Table 2: SQuAD 1.1 results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	86.3	89.0	86.9	89.5
#1 Single - MIR-MRC (F-Net)	-	-	74.8	78.0
#2 Single - nlnet	-	-	74.2	77.1
Published				
unet (Ensemble)	-	-	71.4	74.9
SLQA+ (Single)	-		71.4	74.4
Ours				
$BERT_{LARGE}$ (Single)	78.7	81.9	80.0	83.1

Table 3: SQuAD 2.0 results. We exclude entries that use BERT as one of their components.

System	Dev	Test
ESIM+GloVe	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
$BERT_{BASE}$	81.6	-
$BERT_{LARGE}$	**86.6**	**86.3**
Human (expert)†	-	85.0
Human (5 annotations)†	-	88.0

Table 4: SWAG Dev and Test accuracies. †Human performance is measured with 100 samples, as reported in the SWAG paper.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Ablation Studies

System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

BERT is effective for both fine-tuning and feature-based approaches

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Ablation Studies

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

The deeper model, the better generalization

	Dev Set				
Tasks	MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

Pre-training Tasks matters

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Summary

- Based on Transformer, BERT is a **fast** and **bidirectional pre-trained** model for NLP tasks
- Training BERT includes 2 steps:
  - Pretraining: use **self-supervised** techniques to build good representation space
  - Fine-tuning: make use pre-trained representation for downstream tasks
- BERT archives SOTA across many tasks:
  - Proving its **context understanding** in NLP
  - Showing a good pre-trained encoder for downstream tasks

Table of contents (Reviewers)

1. Brief Summary of BERT
2. Reviewer Comments
3. Conclusion and Discussion

The aim of peer review is to provide authors with constructive feedback from subject experts, so that they can make improvements to their manuscript.

Legends

Positive Point
Critical Point

🔎: Sachin Kumar

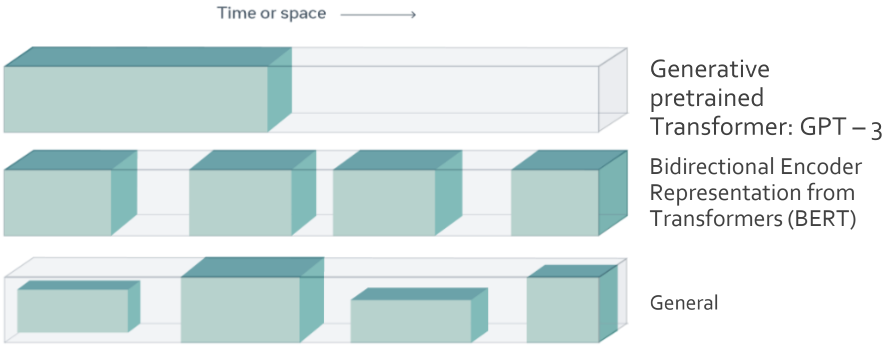
Brief Summary of BERT

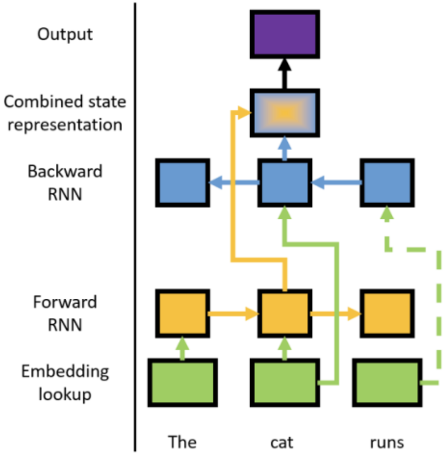
What is BERT?

A predictive language Model that takes into account bi-directional context.

How ?

Masked Language Modelling



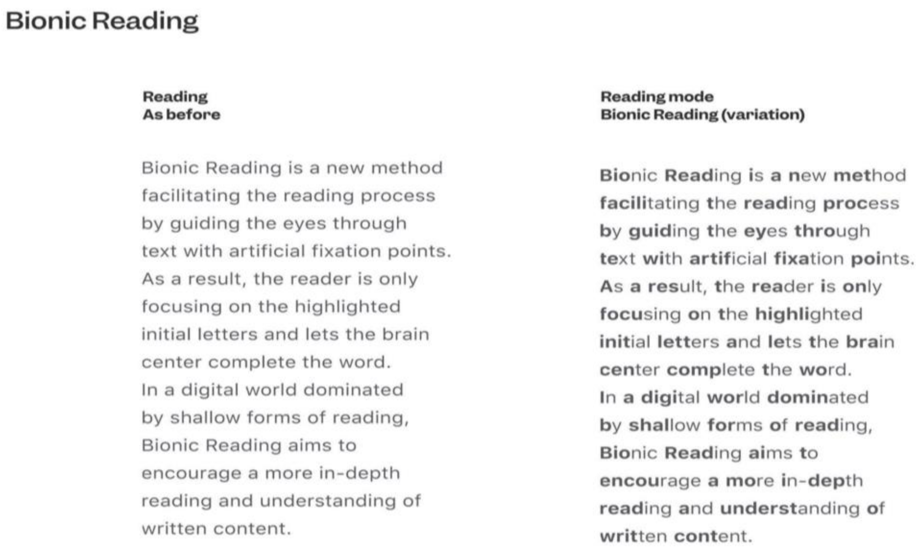


Problem in defining bi-directional context with models that are defined to predict next given past.

Reviewers Comments

Background

- The research indicates brain reads faster when pseudo masked
- UNIDIRECTIONAL!!



https://bionic-reading.com/

Comments

- Hence, BERT is loosely doing something similar to how brain does it.
- BUT it used LTR and RTL?
- Does our brain look at the future context while understanding language?

🔎: Sachin Kumar

Just, Marcel Adam and Patricia A. Carpenter. "A theory of reading: from eye fixations to comprehension." *Psychological review* 87 4 (1980): 329-54 .

Reviewers Comments

- 🟡 BERT trained on the BooksCorpus, a much larger pretraining corpus than GPT and ELMo (their baselines). Why not compare on equal grounds?

- 🟡 Pretraining is a resource intensive process – how can others reproduce your results?

Reviewers Comments

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**

Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

Subset: wnli
Split: train

sentence1 (string)	sentence2 (string)
I stuck a pin through a carrot. When I pulled the pin out, it had a hole.	The carrot had a hole.

[8]See (10) in https://gluebenchmark.com/faq.

12. I get weird results for QQP or WNLI. What gives?

QQP: There is a difference in the dev and test distributions that likely explains discrepancies observed between scores for the two. WNLI: The train/dev split for WNLI is correct, but turns out to be somewhat adversarial: when two examples contain the same sentence, that usually means they'll have opposite labels. The train and dev splits may share sentences, so if a model has overfit the training set, it may get worse than chance accuracy on WNLI on the dev set. Additionally, the test set has a different label distribution than the train and dev sets.

- Overall, BERT shows great improvements over all the baselines

- BUT curious as to why BERT never mentioned WNLI task results.
  - they claim based on the FAQs that WNLI did not perform well because of the dataset mismatch BUT they mention QQP.
  - Curious about the LM performance on the WNLI task. Is the bi-directional context confusing the model for the WNLI?

: Sachin Kumar

Reviewers Comments

- Why not a more contextually heavy task such as the Argument Reasoning Comprehension Task(ARCT)

Unit	Text
Reason	Cooperating with Russia on terrorism ignores Russia's overall objectives.
Claim	Russia cannot be a partner.
Warrant0	Russia has the same objectives of the US.
Warrant1	Russia has the opposite objectives of the US.
Reason	Economic growth needs innovation.
Claim	3-D printing will change the world.
Warrant0	There is no innovation in 3-d printing since it's unsustainable.
Warrant1	There is much innovation in 3-d printing and it is sustainable.
Reason	College students have the best chance of knowing history.
Claim	College students' votes do matter in an election.
Warrant0	Knowing history doesn't mean that we will repeat it.
Warrant1	Knowing history means that we won't repeat it.

ARCT snippet

: Sachin Kumar

Reviewers Comment

word-piece tokenizer concept

: Sachin Kumar

Reviewers Comments

- Over parameterized and no analysis on the inference time
- Effects of Increase/decrease in number of attention heads and its effects on the accuracy of the NLP tasks.

Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Table 6: Ablation over BERT model size. #L = the number of layers; #H = hidden size; #A = number of attention heads. "LM (ppl)" is the masked LM perplexity of held-out training data.

: Sachin Kumar

Conclusion (Gist of other Comments)

High Performance.	Very compute Intensive.
Truly bidirectional context	Unfair comparison to other baselines It is slow to train because it is big and there are a lot of parameters to update.
The objectives have theoretical foundations in how humans learn	Certain critical tasks like WNLI are ignored
The tokenizer makes the vocabulary open	Limited analysis

🔎: Sachin Kumar

Journey of BERT

Last week…

- What problem was RNN trying to solve?

Last week…

- What problem was RNN trying to solve?
    - (Conditional) Language Model

Last week…

- What problem was RNN trying to solve?
  - Conditional Language Modeling
- What were the issues with Recurrent Neural Networks?

Last week…

- What problem was RNN trying to solve?
  - Conditional Language Modeling
- What were the issues with Recurrent Neural Networks?
  - "Recurrent computation is slow"
  - Long sequences could result in parts of the input being forgotten.

What Inspired BERT?

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Illia Polosukhin* ‡ illia.polosukhin@gmail.c

Deep contextualized word representations

Matthew E. Peters[†], Mark Neumann[†], Mohit Iyyer[†], Matt Gardner[†],
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark[*], Kenton Lee[*], Luke Zettlemoyer[†*]
{csquared,kentonl,lsz}@cs.washington.edu

[†]Allen Institute for Artificial Intelligence
[*]Paul G. Allen School of Computer Science & Engineering, University of Washing

Universal Language Model Fine-tuning for Text Classification

Jeremy Howard[*]
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder[*]
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io

Timeline

LONG SHORT-TERM MEMORY

NEURAL COMPUTATION 9(8):1735–1780, 1997

Sepp Hochreiter
Fakultät für Informatik
Technische Universität München
80290 München, Germany
hochreit@informatik.tu-muenchen.de
http://www7.informatik.tu-muenchen.de/~hochreit

Jürgen Schmidhuber
IDSIA
Corso Elvezia 36
6900 Lugano, Switzerland
juergen@idsia.ch
http://www.idsia.ch/~juergen

01 ELMo, ULMFit

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

02 **Transformer**

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

03 BERT

ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS

Zhenzhong Lan[1] Mingda Chen[2*] Sebastian Goodman[1] Kevin Gimpel[2]
Piyush Sharma[1] Radu Soricut[1]
[1]Google Research [2]Toyota Technological Institute at Chicago
{lanzhzh, seabass, piyushsharma, rsoricut}@google.com
{mchen, kgimpel}@ttic.edu

05 ALBERT, T5 and GPT2

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Colin Raffel* CRAFFEL@GMAIL.COM
Noam Shazeer* NOAM@GOOGLE.COM
Adam Roberts* ADAROB@GOOGLE.COM
Katherine Lee* KATHERINELEE@GOOGLE.COM
Sharan Narang SHARANNARANG@GOOGLE.COM
Michael Matena MMATENA@GOOGLE.COM
Yanqi Zhou YANQIZ@GOOGLE.COM
Wei Li MWEILI@GOOGLE.COM
Peter J. Liu PETERJLIU@GOOGLE.COM
Google, Mountain View, CA 94043, USA

Language Models are Unsupervised Multitask Learners

Alec Radford * [1] Jeffrey Wu * [1] Rewon Child [1] David Luan [1] Dario Amodei ** [1] Ilya Sutskever ** [1]

1735-1780, 1997 **2017** **2019** **2020**

ALBERT: A Lite BERT

- Why ALBERT
- How ALBERT works
- Performance ALBERT v.s. BERT

Why ALBERT

- The problems in BERT:
  - Memory limitation
    - Model parallelization ✅
    - Clever management ✅
  - Communication overhead
    - ALBERT incorporates 2 parameter reduction techniques:
      - Factorized embedding parameterization
      - Cross layer parameter sharing
  - Next Sentence Prediction (NSP) ineffectiveness
    - Self-supervised loss for sentence-order prediction (SOP)

How ALBERT works

- Factorized embedding parameterization
  - Recall BERT
    - Embedding Size E = Hidden Layer Size H
  - Question:
    - E: context independent
    - H: context dependent
  - Reduce Embedding Parameters
    - First project one-hot vectors into a lower dimensional embedding size E
    - Then project it into hidden space
    - $O(V*H)$ ➡️ $O(V*E+E*H)$
    - E: 64, 128(best), 256, 768

How ALBERT works

- Cross-layer parameter sharing
    - Share all parameters across layers
    - Prevent the parameter from growth with the depth of network
    - Weight-sharing has an effect on stabilizing network parameters

How ALBERT works

- Inter-sentence coherence loss
  - Why NSP ineffectiveness
    - Lack of difficulty as a task
      - NSP conflates topic prediction and coherence prediction in a single task
      - Topic prediction is much easier
  - ALBERT: sentence order prediction (SOP) loss
    - Avoid topic prediction
    - Focuses on modeling inter-sentence coherence

Performance ALBERT v.s. BERT Factorized embedding parameterization

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.4/83.2	80.4/77.6	84.5	92.8	68.2	82.3	4.7x
	large	334M	92.2/85.5	85.0/82.2	86.6	93.0	73.9	85.2	1.0
ALBERT	base	12M	89.3/82.3	80.0/77.1	81.6	90.3	64.0	80.1	5.6x
	large	18M	90.6/83.9	82.3/79.4	83.5	91.7	68.5	82.4	1.7x
	xlarge	60M	92.5/86.1	86.1/83.1	86.4	92.4	74.8	85.5	0.6x
	xxlarge	235M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	0.3x

Table 2: Dev set results for models pretrained over BOOKCORPUS and Wikipedia for 125k steps. Here and everywhere else, the Avg column is computed by averaging the scores of the downstream tasks to its left (the two numbers of F1 and EM for each SQuAD are first averaged).

Performance ALBERT v.s. BERT
Cross-layer parameter sharing

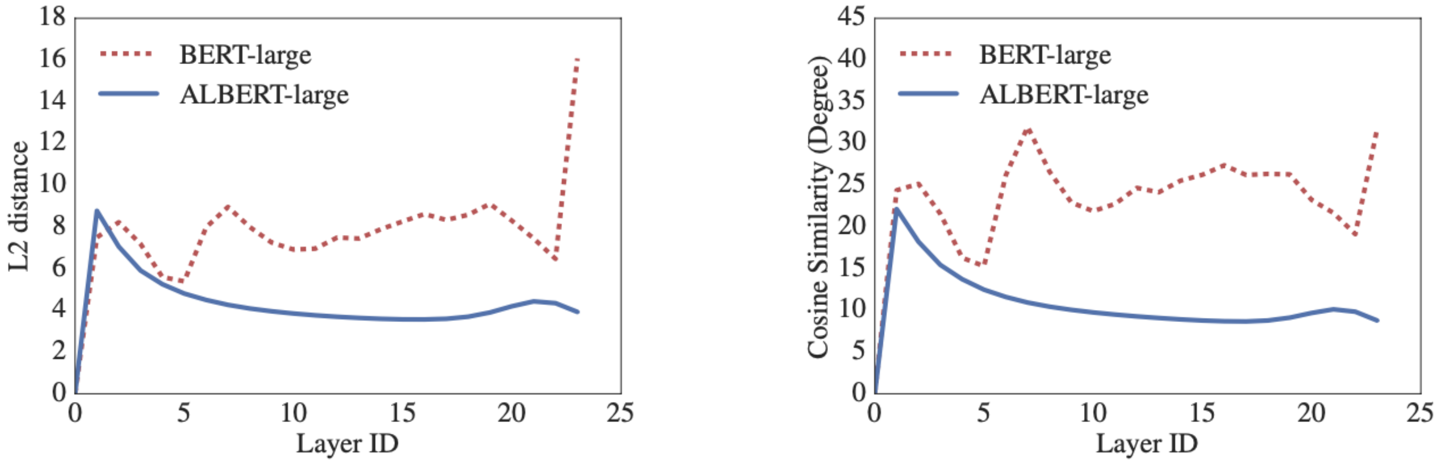
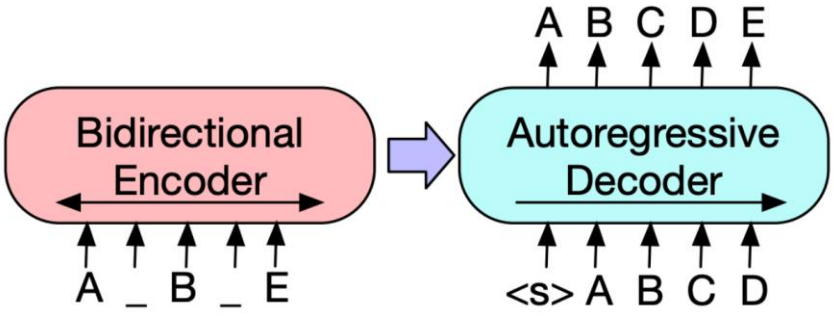


Figure 1: The L2 distances and cosine similarity (in terms of degree) of the input and output embedding of each layer for BERT-large and ALBERT-large.

T5

T5: Main Idea

- Encoder Decoder Variant of BERT
  - Encoder Input: Masked Sequence
  - Decoder Output: Full unmasked sequence

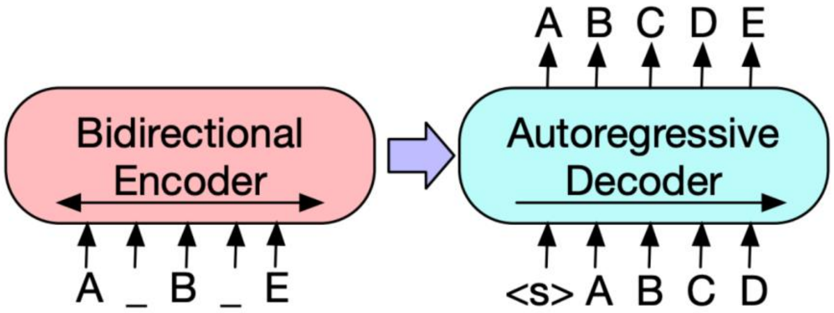


- Why?
  - More Flexibility
  - Can easily finetune for sequence to sequence tasks like translation and summarization.

Pretraining Data

- Colossal Cleaned Common Crawl: 156B tokens compared to 33B for BERT

- Sizes similar to BERT



Language Models are Unsupervised Multitask Learners (GPT2)

GPT-2: Main Idea

- Train a unidirectional language model with a next-token prediction objective (the OG language model)
    - Also known as causal or autoregressive language models

- Use case: same as BERT/T5 but focused on generating text
    - But T5 can also generate text
    - Yes, but it is trained to denoise, not as a language model
    - Can we train a T5 like model with a next token prediction objective
        - Yes—check prefix LM

GPT-2: Why?

- Much simpler pretraining objective than masked/denoising LMs – way more sample efficient, easier to scale
    - Largest size of BERT-like models (less than 1B) << Largest size of GPT like models (>500B)

- Works for several tasks *without* finetuning
    - **Zero shot** capabilities

GPT-2: Zero-shot capabilities

- GPT-2 achieves state-of-the-art scores on a variety of domain-specific language modeling tasks (perplexity).

Dataset	Metric	Our result	Previous record	Human
Winograd Schema Challenge	accuracy (+)	70.70%	63.7%	92%+
LAMBADA	accuracy (+)	63.24%	59.23%	95%+
LAMBADA	perplexity (–)	8.6	99	~1–2
Children's Book Test Common Nouns (validation accuracy)	accuracy (+)	93.30%	85.7%	96%
Children's Book Test Named Entities (validation accuracy)	accuracy (+)	89.05%	82.3%	92%
Penn Tree Bank	perplexity (–)	35.76	46.54	unknown
WikiText-2	perplexity (–)	18.34	39.14	unknown
enwik8	bits per character (–)	0.93	0.99	unknown
text8	bits per character (–)	0.98	1.08	unknown
WikiText-103	perplexity (–)	17.48	18.3	unknown

GPT-2: Zero-shot capabilities

- We can generate from GPT-2 by sampling from its underlying distribution

- One of the first models to show highly fluent outputs

System Prompt (human-written)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Model Completion (machine-written, 10 tries)

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

Pérez and the others then ventured further into the valley. "By the time we reached the top of one peak, the water looked blue, with some crystals on top," said Pérez.

Pérez and his friends were astonished to see the unicorn herd. These creatures could be seen from the air without having to move too much to see them – they were so close they could touch their horns.

Controversy

Release strategy

Due to concerns about large language models being used to generate deceptive, biased, or abusive language at scale, we are only releasing a much smaller version of GPT-2 along with sampling code. We are not releasing the dataset, training code, or GPT-2 model weights. Nearly a year ago we wrote in the OpenAI Charter: "we expect that safety and security concerns will reduce our traditional publishing in the future, while increasing the importance of sharing safety, policy, and standards research," and we see this current work as potentially representing the early beginnings of such concerns, which we expect may grow over time. This decision, as well as our discussion of it, is an experiment: while we are not sure that it is the right decision today, we believe that the AI community will eventually need to tackle the issue of publication norms in a thoughtful way in certain research areas. Other disciplines such as biotechnology and cybersecurity have long had active debates about responsible publication in cases with clear misuse potential, and we hope that our experiment will serve as a case study for more nuanced discussions of model and code release decisions in the AI community.

Exploration: How do we make the models smaller ?

- Post Training: Are all model parameters effectively getting utilized? Exploring ideas related to pruning neural networks

- Training from Scratch: Can we use knowledge distillation?
  - Student-Teacher training where a teacher network adds its error to the student's loss function, thus, helping the student network to converge to a better solution.

Exploration: Does pretraining work well in other languages?

- Train BERT/GPT2 in languages from different families and writing scripts. Compare performance differences?
    - How much data is needed to achieve good performance?

- Train a multilingual model capable of working in multiple languages at the same time.

Exploration: Can masked prediction be applied to other modalities?

- Train BERT/GPT2 in languages from different families and writing scripts. Compare performance differences?

- Train a multilingual model capable of working in multiple languages at the same time.

Questions?

Logistics - FQA

- How many papers in total do I need to present throughout the semester?

Logistics - FQA

- How many papers in total do I need to present throughout the semester?

  Each student will present for each role once.

Logistics - FAQ

- What if I am presenting but having trouble understanding some parts of the paper? Will I get penalized?

Logistics - FAQ

- What if I am presenting but having trouble understanding some parts of the paper? Will I get penalized?
  - You are not the author of the paper. It is okay if you don't completely understand every detail!
  - We will try to understand the details in discussions
  - Also feel free to reach out to ask questions