AIRI × Университет Сириус

Attention!
Transformers

Aydar Bulatov

Feel free to open this lecture on your laptop

Telegram

Github of NLP Course

Feedback

Today

01 Quick recap

02 Attention mechanism

03 Applications and variations

Let’s look back

How do we start solving any NLP problem?

Processing language

We’ve talked about:

1. Count-based methods

   Co-occurrence counts, Tf-Idf

2. Prediction-based methods

   Word2Vec, GloVe, FastText

3. RNN, LSTM

What do they have in common?

Processing language

We’ve talked about:

1. Count-based methods

   Co-occurrence counts, Tf-Idf

2. Prediction-based methods

   Word2Vec, GloVe, FastText

3. RNN, LSTM

What do they have in common?

They rely on **word representations**

Limitations

Lack of context

What is a plant?

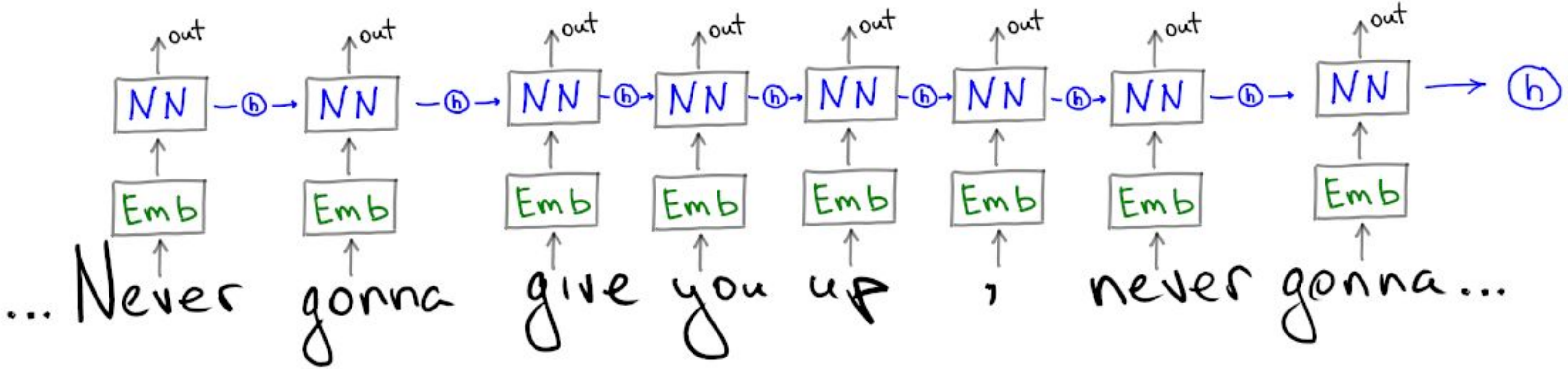


Bottleneck in RNN/LSTM state

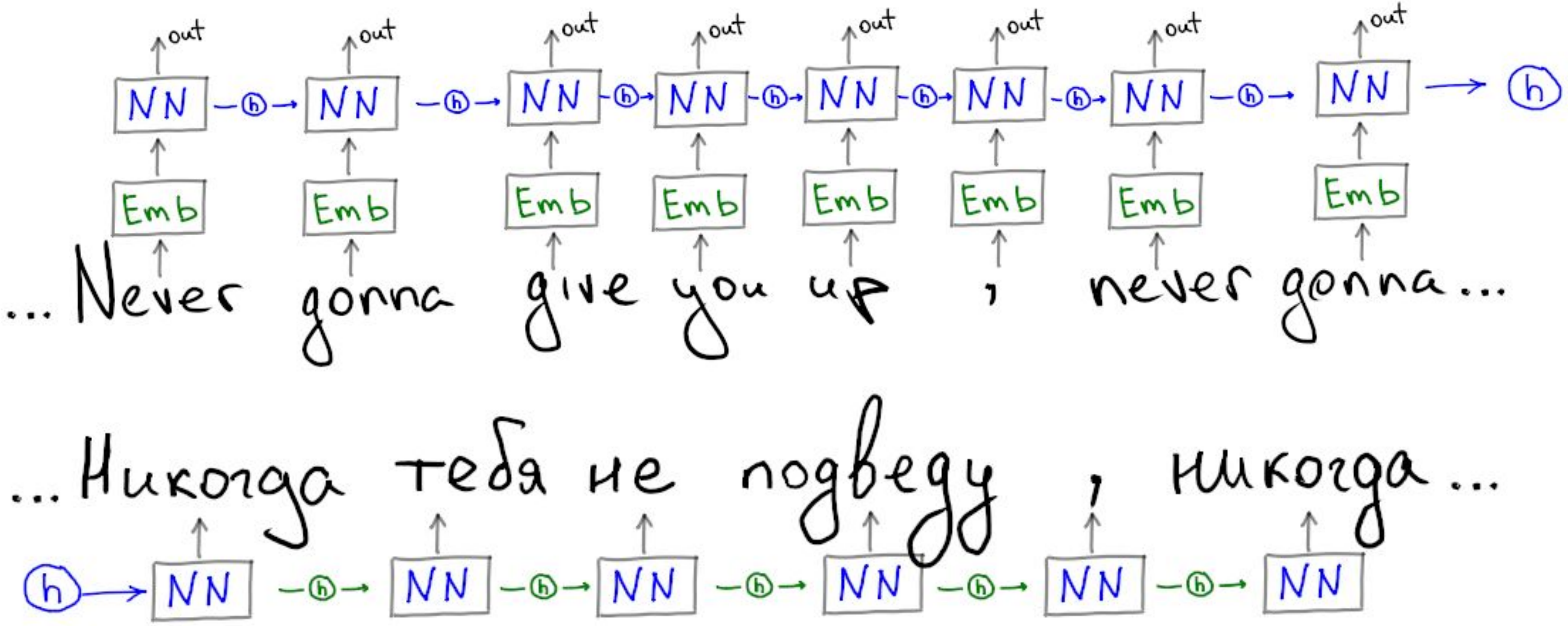
...Never gonna give you up, never gonna...

How would an RNN translate it?

Bottleneck in RNN/LSTM state

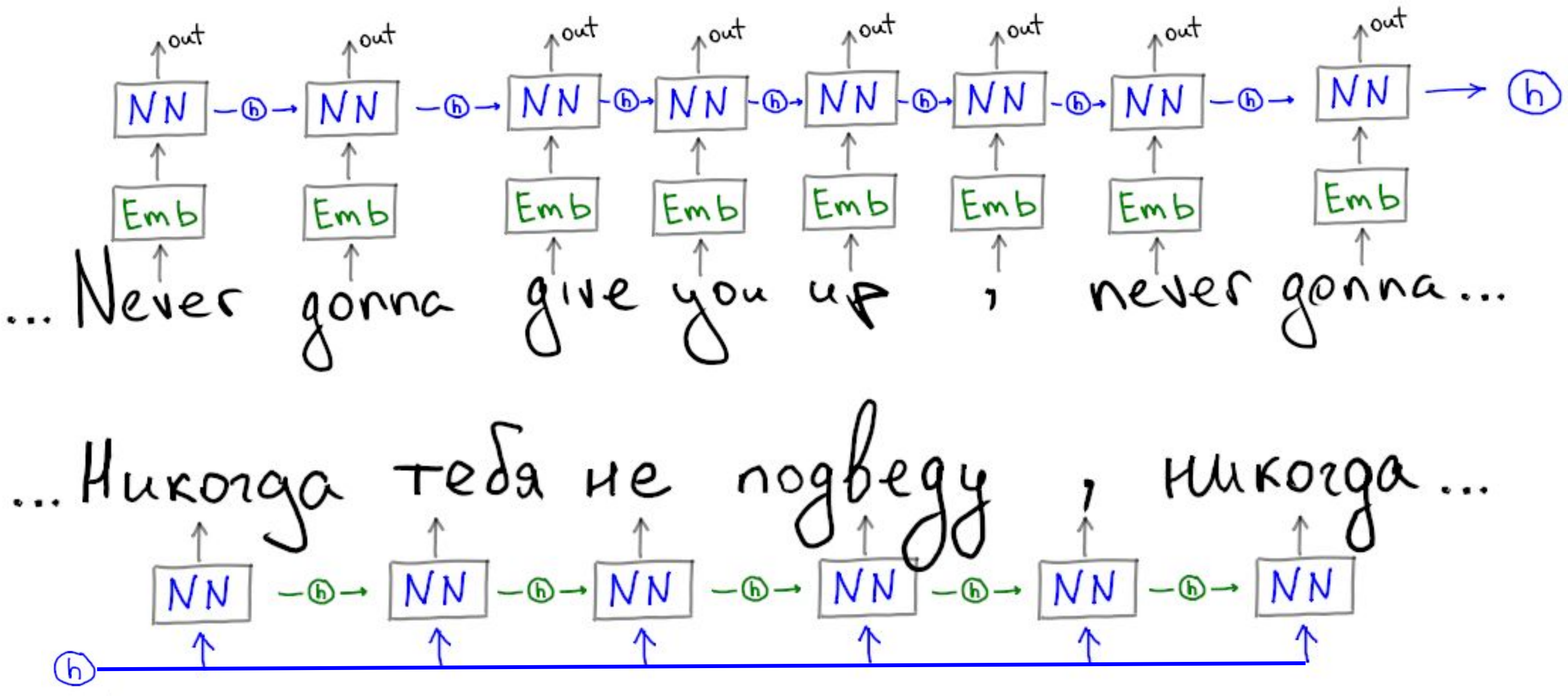


Bottleneck in RNN/LSTM state



AIRI × Университет Сириус

Bottleneck in RNN/LSTM state



Bottleneck in RNN/LSTM state

1) only information we have is a hidden state vector

2) we need to process input sequentially

Bottleneck in RNN/LSTM state

How would **we** do it?

	Never	gonna	give	you	up	,	never	gonna
Никогда	0,9	0,1						
тебя				1				
не	1							
подведу			0,5		0,5			
,						1		
никогда							0,9	0,1

Bottleneck in RNN/LSTM state

Let’s make this matrix trainable

<table>
<tr><th></th><th>Never</th><th>gonna</th><th>give</th><th>you</th><th>up</th><th>,</th><th>never</th><th>gonna</th></tr>
<tr><td>Никогда</td><td colspan="8" rowspan="6">wij</td></tr>
<tr><td>тебя</td></tr>
<tr><td>не</td></tr>
<tr><td>подведу</td></tr>
<tr><td>,</td></tr>
<tr><td>никогда</td></tr>
</table>

Attention

What do we want it to be like?

1) measures compatibility
2) outputs weighted average
3) runs in parallel

Background

A **key–value database**, or key–value store, is a data storage paradigm designed for storing, retrieving, and managing associative arrays.

E.g.: ArangoDB, MongoDB, etc.

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

Background

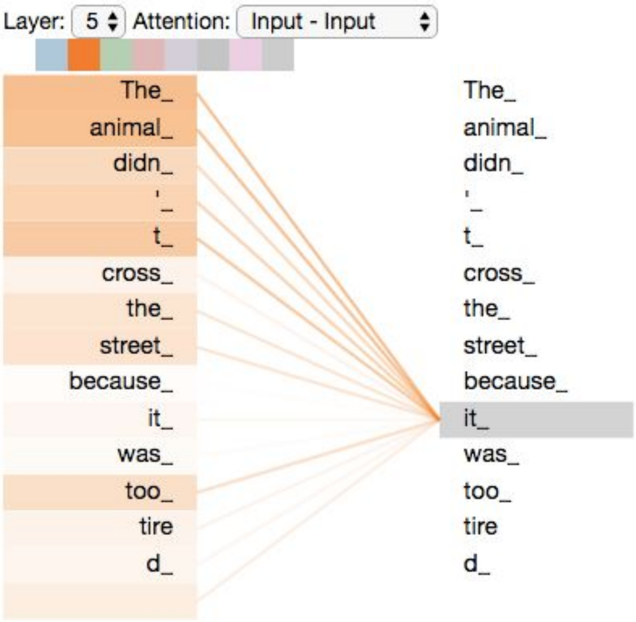
A **key–value database**, or key–value store, is a data storage paradigm designed for storing, retrieving, and managing associative arrays.

E.g.: ArangoDB, MongoDB, etc.

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

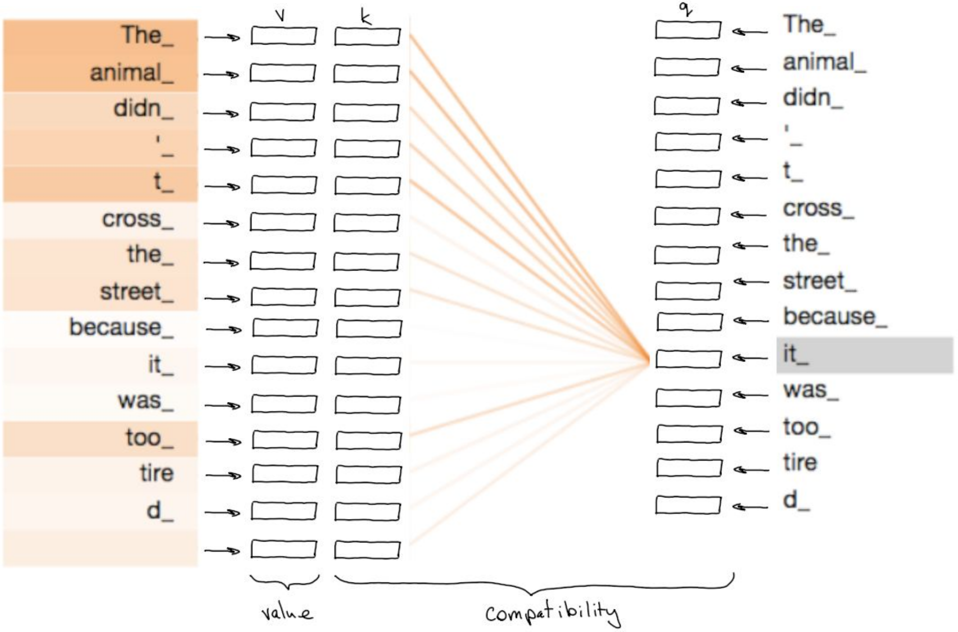
Let's treat our input as a database!

Attention



Takes weighted average

Attention

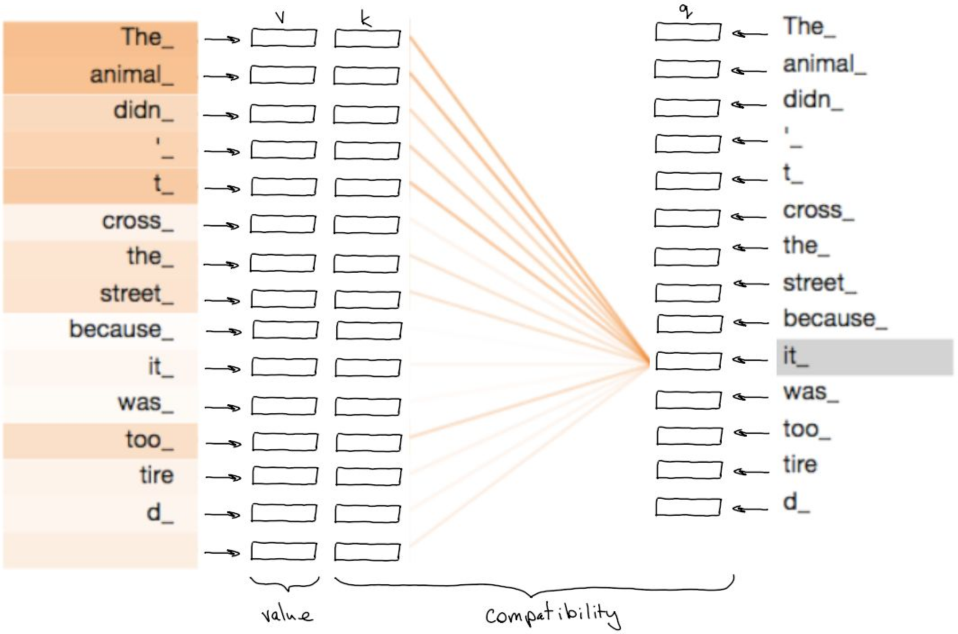


$$\mathit{Attention}(q, k, v) = \overbrace{\mathit{softmax}\left(\frac{qk^T}{\sqrt{d_k}}\right)}^{\text{Attention weights}} v$$

from to

vector dimensionality of K, V

Attention



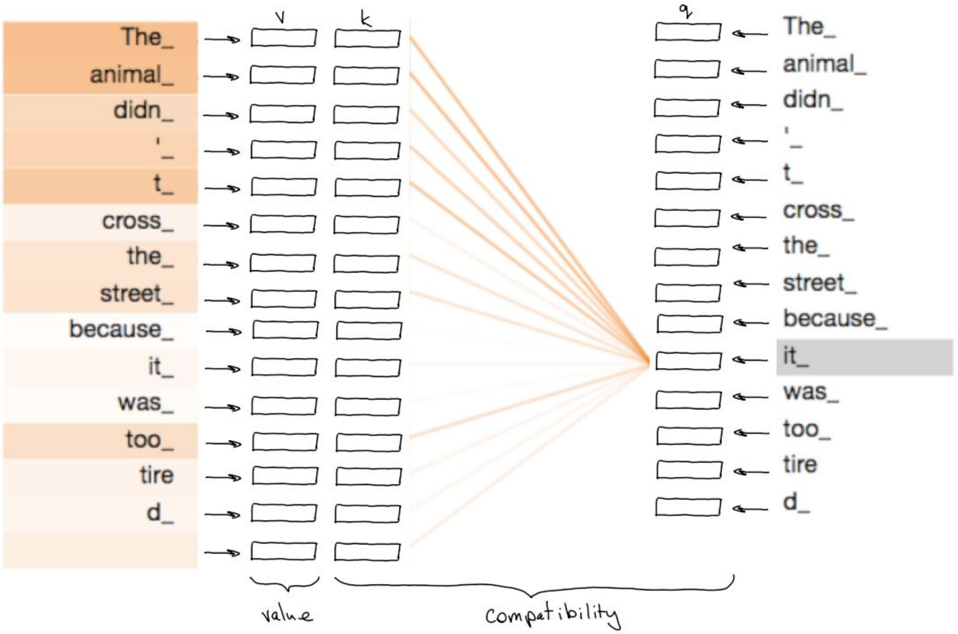
$$Attention(q, k, v) = \overbrace{softmax\left(\frac{qk^T}{\sqrt{d_k}}\right)}^{\text{Attention weights}} v$$

from to

vector dimensionality of K, V

seems familiar...

Attention



$$\text{Attention}(q,k,v) = \overbrace{\text{softmax}\left(\frac{qk^T}{\sqrt{d_k}}\right)}^{\text{Attention weights}} v$$

from   to

vector dimensionality of K, V

seems familiar...

Word2Vec

$$P(o|c) = SM(u_o^T \cdot v_c) = \frac{e^{u_o^T v_c}}{\sum_{w \in V} e^{u_w^T v_c}}$$

Attention

Instead of computing:
Attention(X, xi) for each *xi*

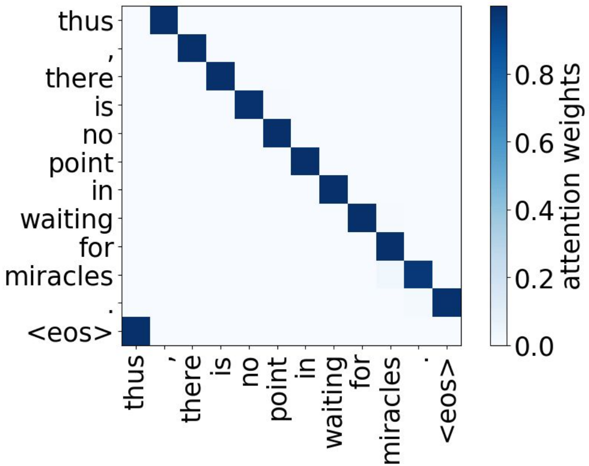
We can compute in parallel:
Attention(X, X)

$$Attention(q, k, v) = \overbrace{softmax\left(\frac{qk^T}{\sqrt{d_k}}\right)}^{\text{Attention weights}} v$$

from to

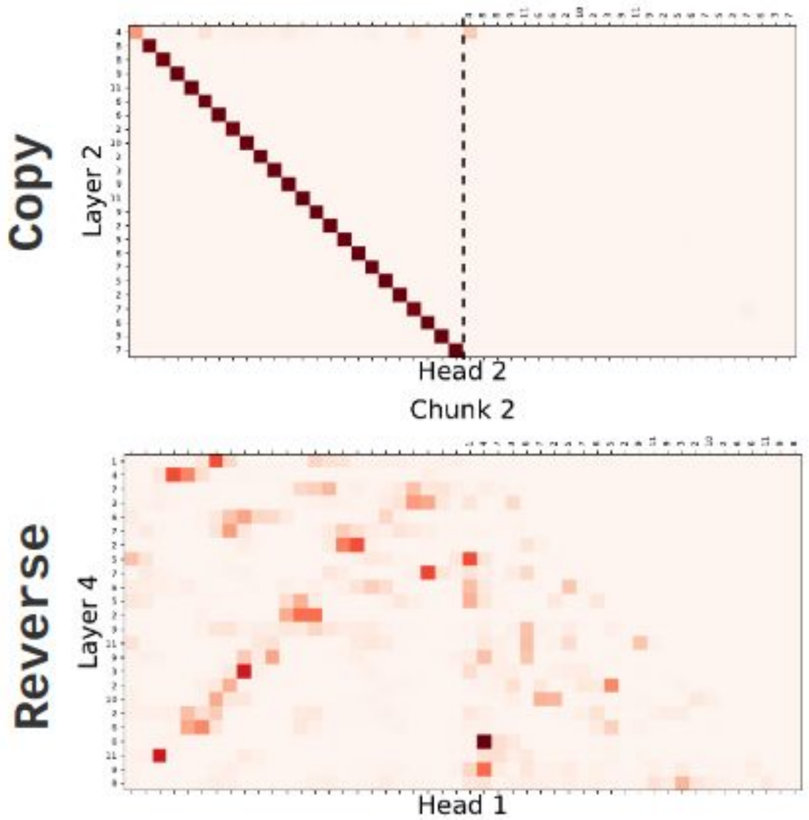
vector dimensionality of K, V

Attention map



What is missing?

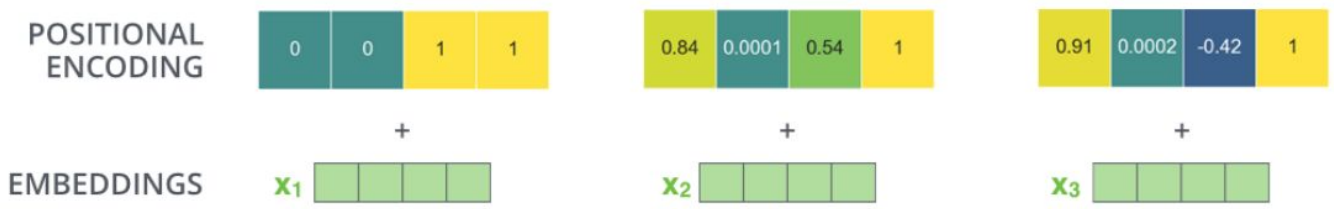
Attention

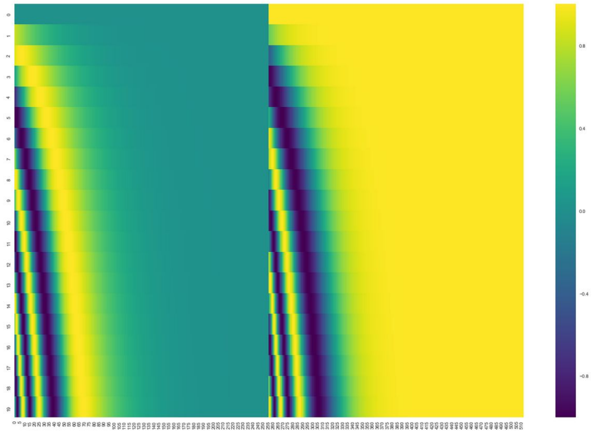


AIRI × Университет Сириус

Positional encoding

We need to represent the sequence order





Attention is all you need

- Introduces Transformer
  encoder-decoder architecture
  based *solely* on attention
- Uses multi-head attention
- Sine and cosine positional encoding
- Achieves SOTA on translation
- With less training cost

Vaswani et al. (2017)

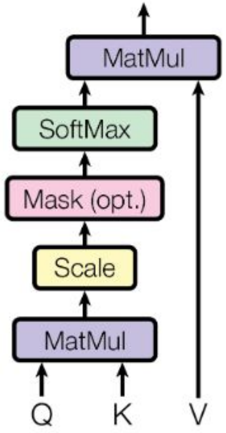
Attention is all you need

- Introduces Transformer encoder-decoder architecture based *solely* on attention
- Uses multi-head attention
- Sine and cosine positional encoding
- Achieves SOTA on translation
- With less training cost

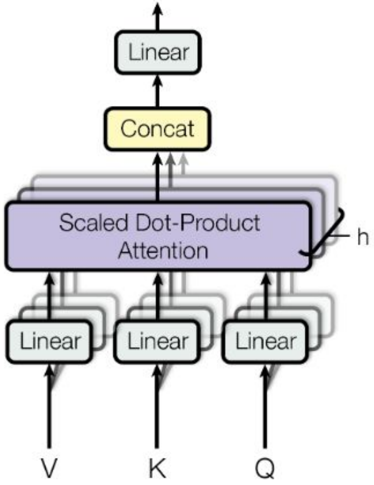
Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
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**	$2.3 \cdot 10^{19}$	

Vaswani et al. (2017)

Attention is all you need



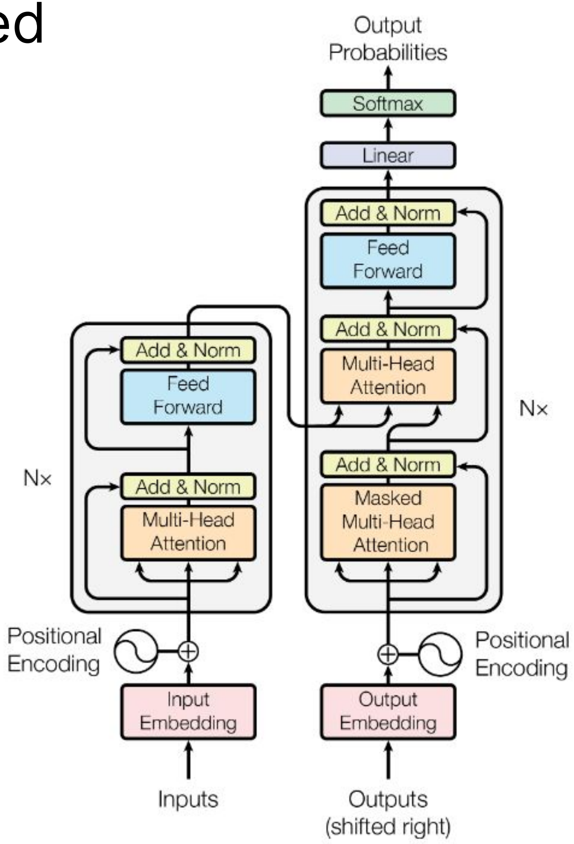
Scaled dot-product attention



Multi-head attention

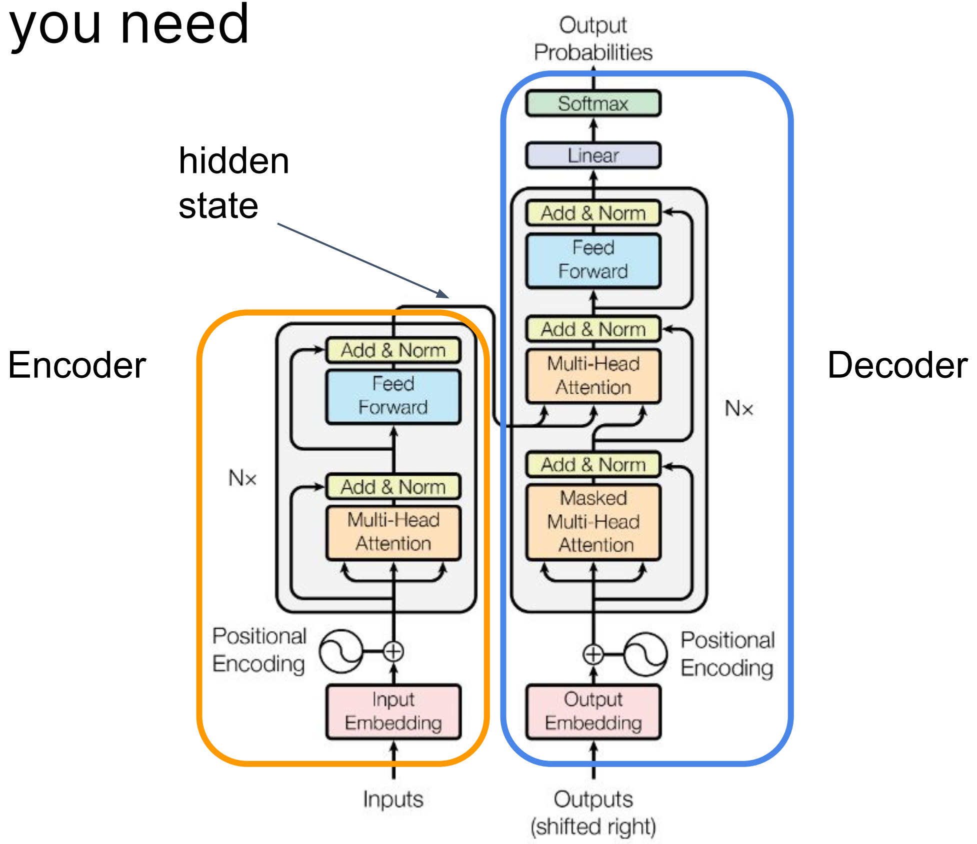
Vaswani et al. (2017)

Attention is all you need



[Vaswani et al. (2017)]

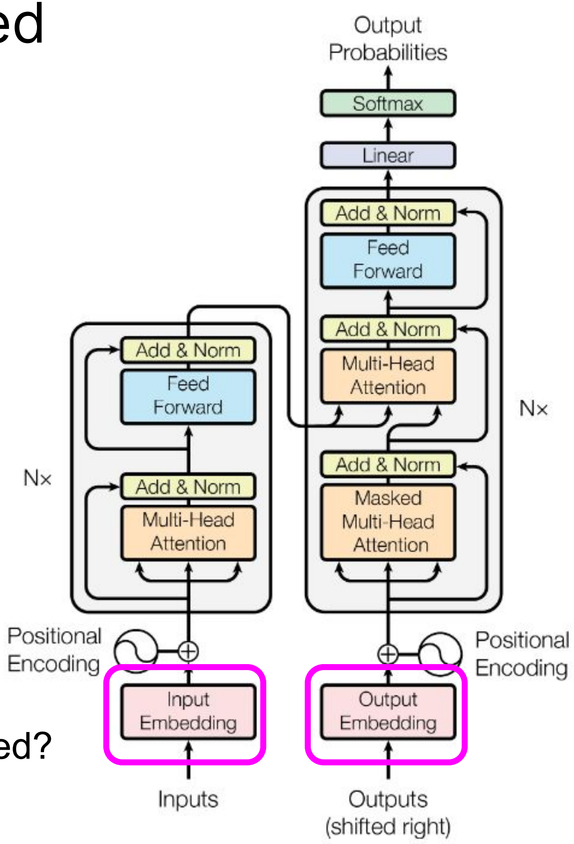
Attention is all you need



[Vaswani et al. (2017)]

AIRI × Университет Сириус

Attention is all you need



What embedding is used?

[Vaswani et al. (2017)]

AIRI × Университет Сириус

BPE

athazagoraphobia = ['_ath', 'az', 'agor', 'aphobia']

Step 0. Initialize vocabulary.

Step 1. Represent each word in the corpus as a combination of the characters along with the special end of word token </w>.

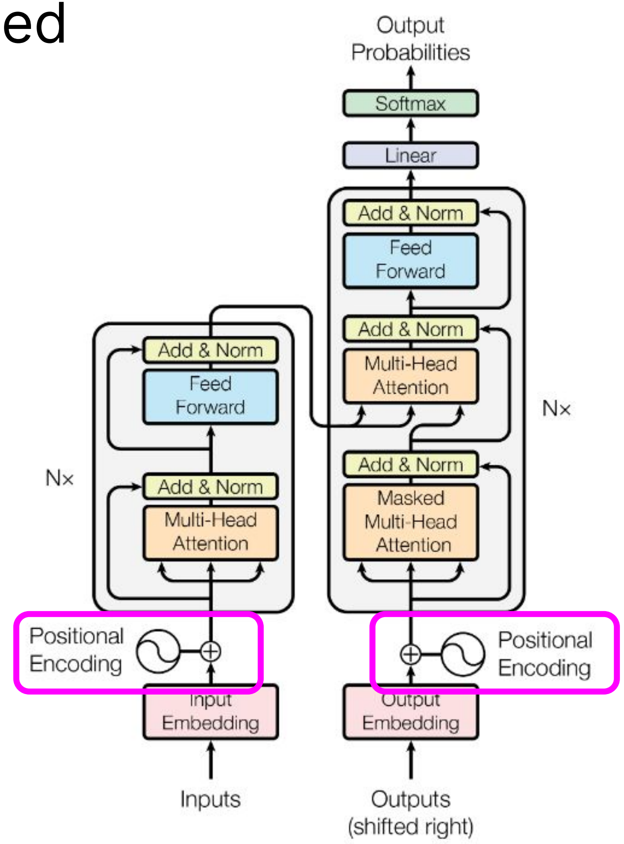
Step 2. Iteratively count character pairs in all tokens of the vocabulary.

Step 3. Merge every occurrence of the most frequent pair, add the new character n-gram to the vocabulary.

Step 4. Repeat step 3 until the desired number of merge operations are completed or the desired vocabulary size is achieved (which is a hyperparameter).

Attention is all you need

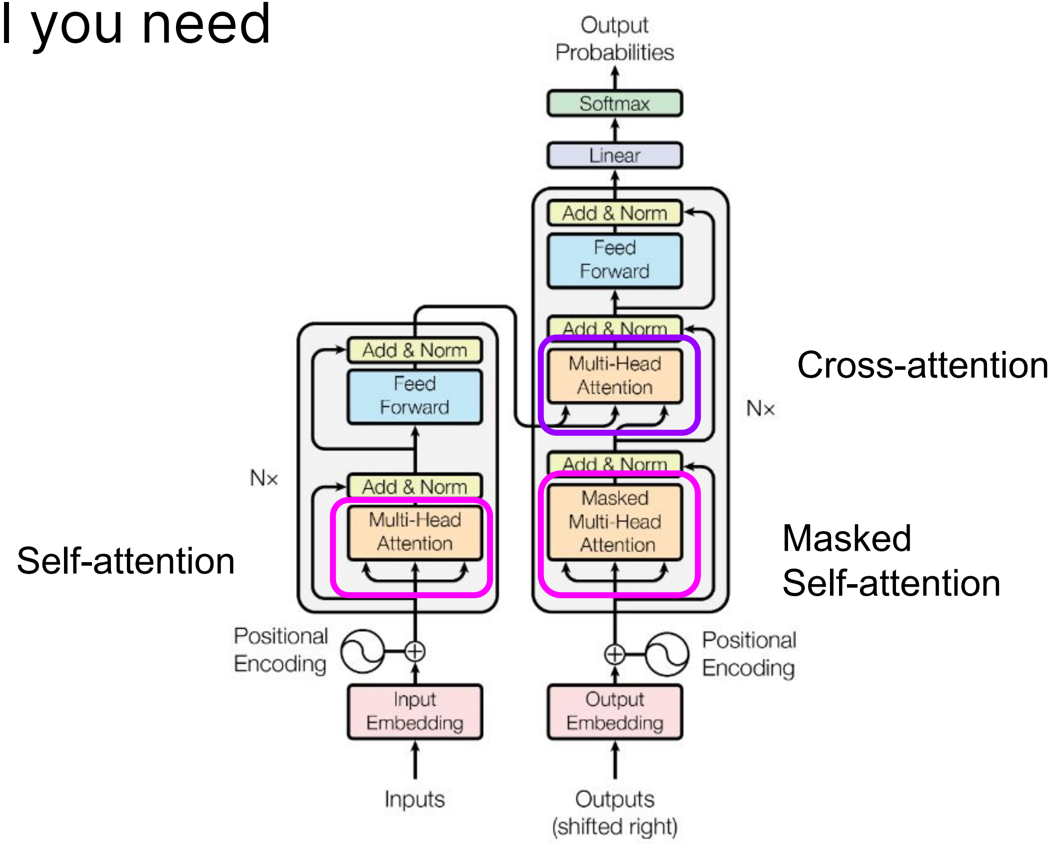
$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$



adds positional information

Vaswani et al. (2017)

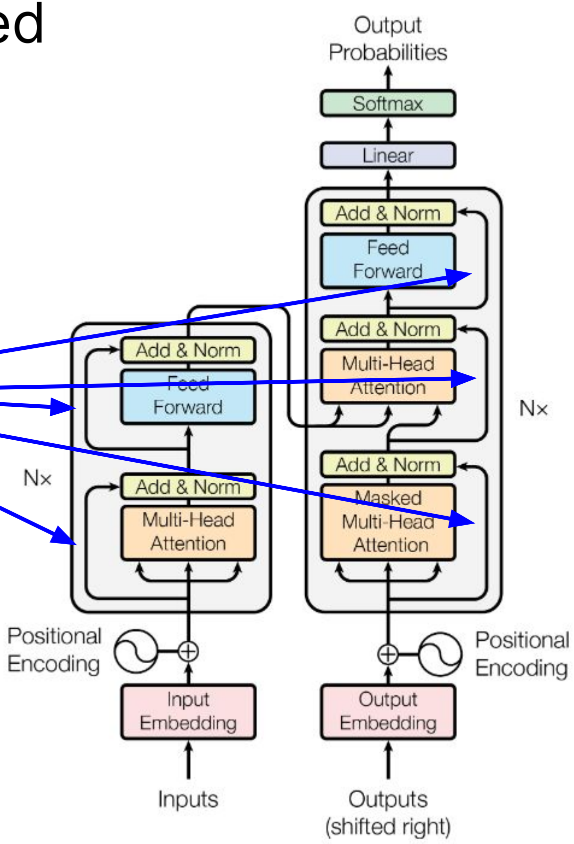
Attention is all you need



Vaswani et al. (2017)

Attention is all you need

Skip-connections

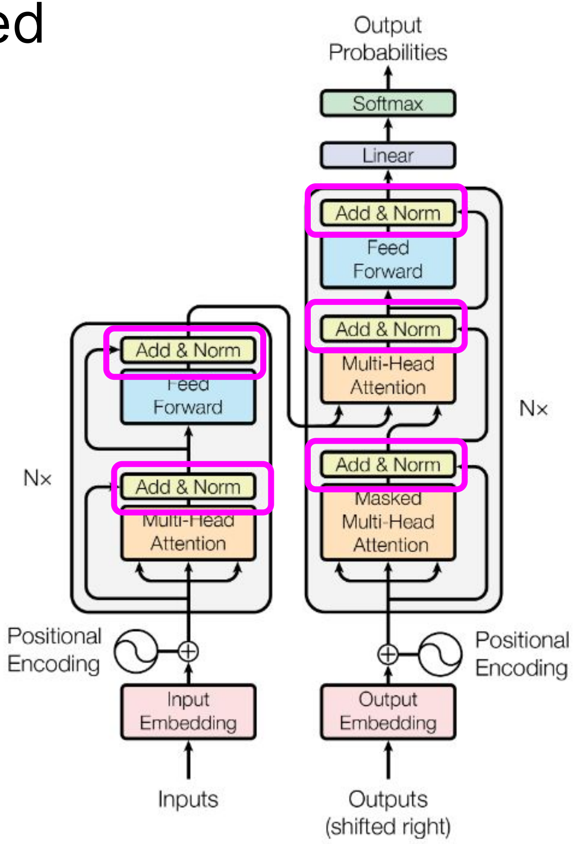


allow gradients to flow freely
speed up training

Vaswani et al. (2017)

Attention is all you need

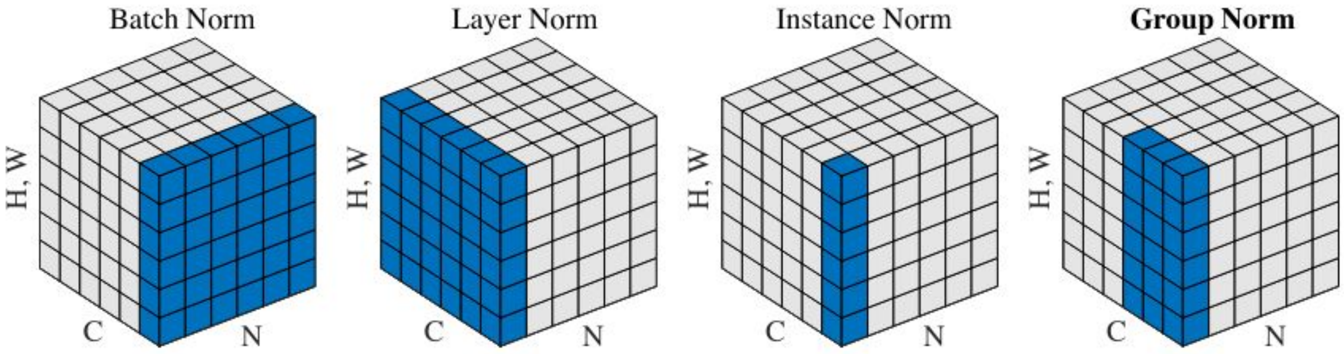
Layer normalization



Vaswani et al. (2017)

Layer Normalization

CV:



Group Normalization

Layer Normalization

NLP:

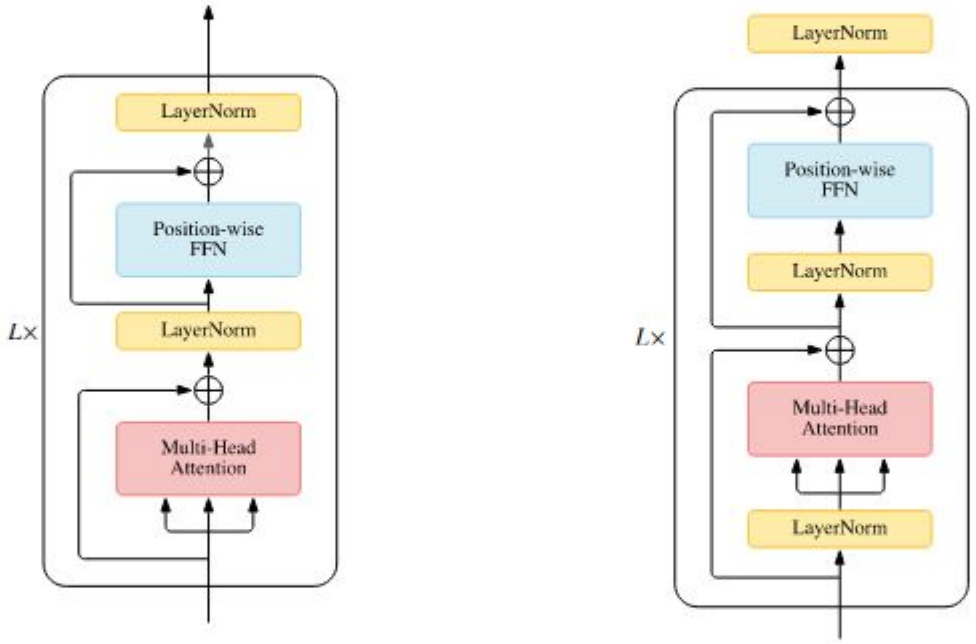
$$\hat{x}_{tdbl} = \frac{x_{tdbl} - \mu_{tdb}}{\sigma}$$

, where

$$\mu_{tdb} = \frac{1}{L} \cdot \sum_{l=1}^{L} x_{tdbl} \qquad \sigma_{tdb} = \sqrt{\frac{1}{L} \cdot \sum_{l=1}^{L} (x_{tdbl} - \mu_{tdb})^2}$$

T - num tokens in sequence, D - embedding dim, B - batch size, L - num layers

Layer normalization



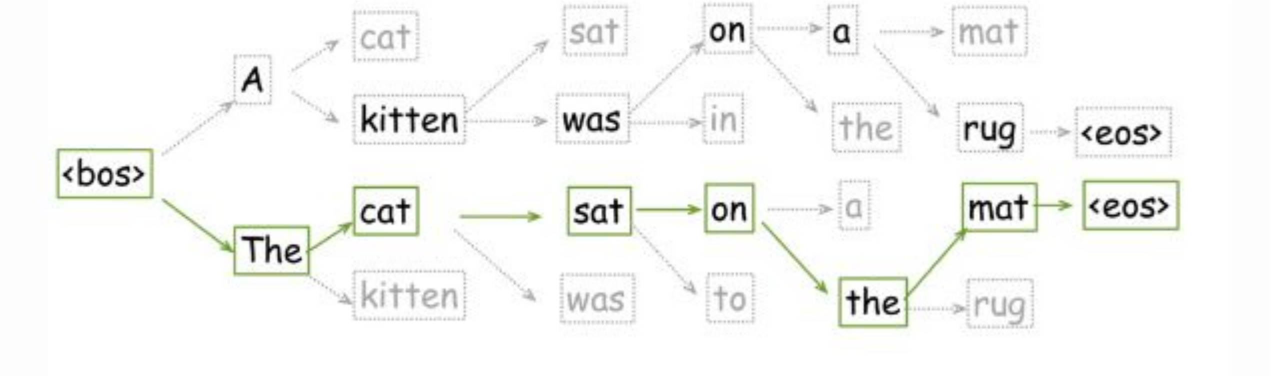
(a) post-LN (b) pre-LN

Ways of decoding

- greedy decoding

$$\arg\max_{y} \prod_{t=1}^{n} p(y_t | y_{<t}, x) \neq \prod_{t=1}^{n} \arg\max_{y_t} p(y_t | y_{<t}, x)$$

- beam search



Where Attention shines

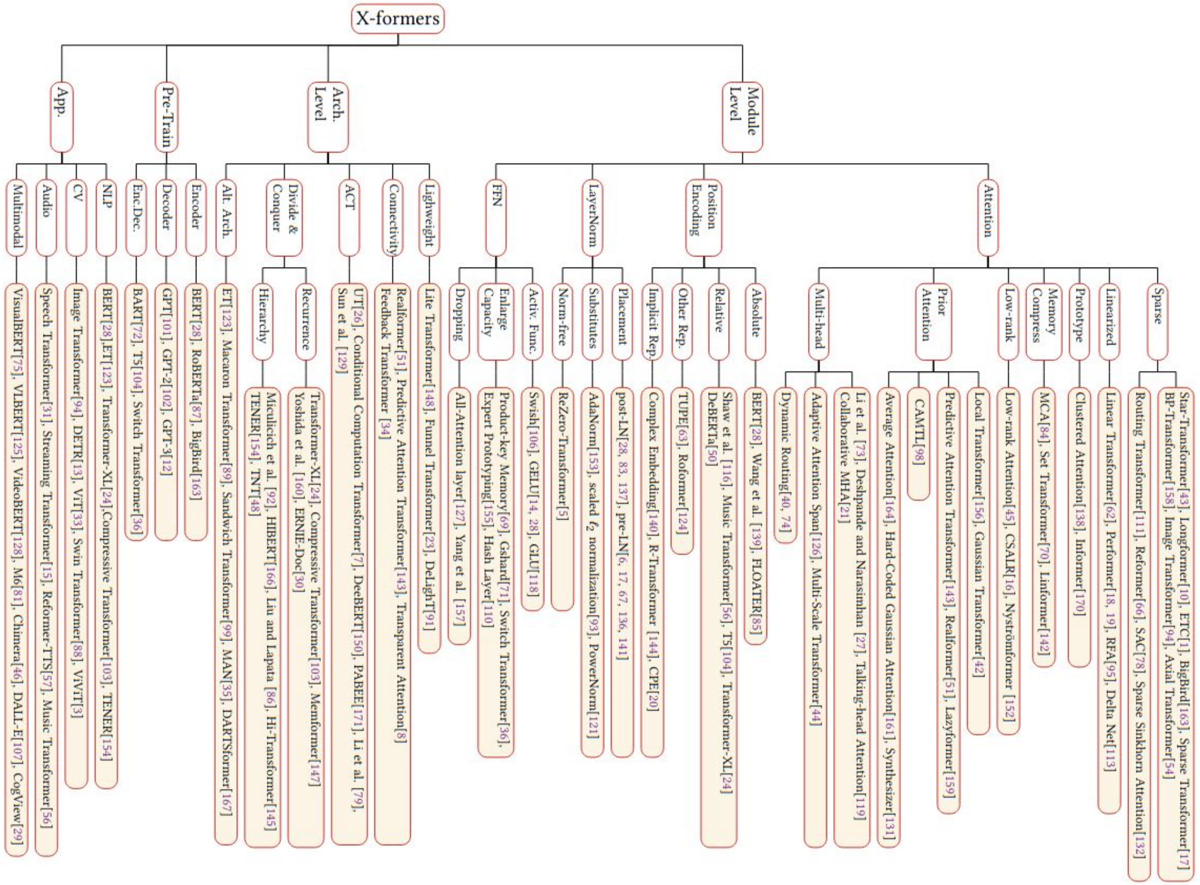
Some applications

- Natural Language Processing
  - machine translation
  - language modeling
  - named entity recognition
  - pre-training on large-scale corpora
- Computer Vision
  - image classification
  - object detection
  - image generation
  - video processing
- Audio
  - speech recognition
  - speech synthesis
  - speech enhancement
  - and music generation
- Multimodal Tasks
  - visual question answering
  - visual common-sense reasoning
  - caption generation
  - speech-to-text translation
  - text-to-image generation

Applications

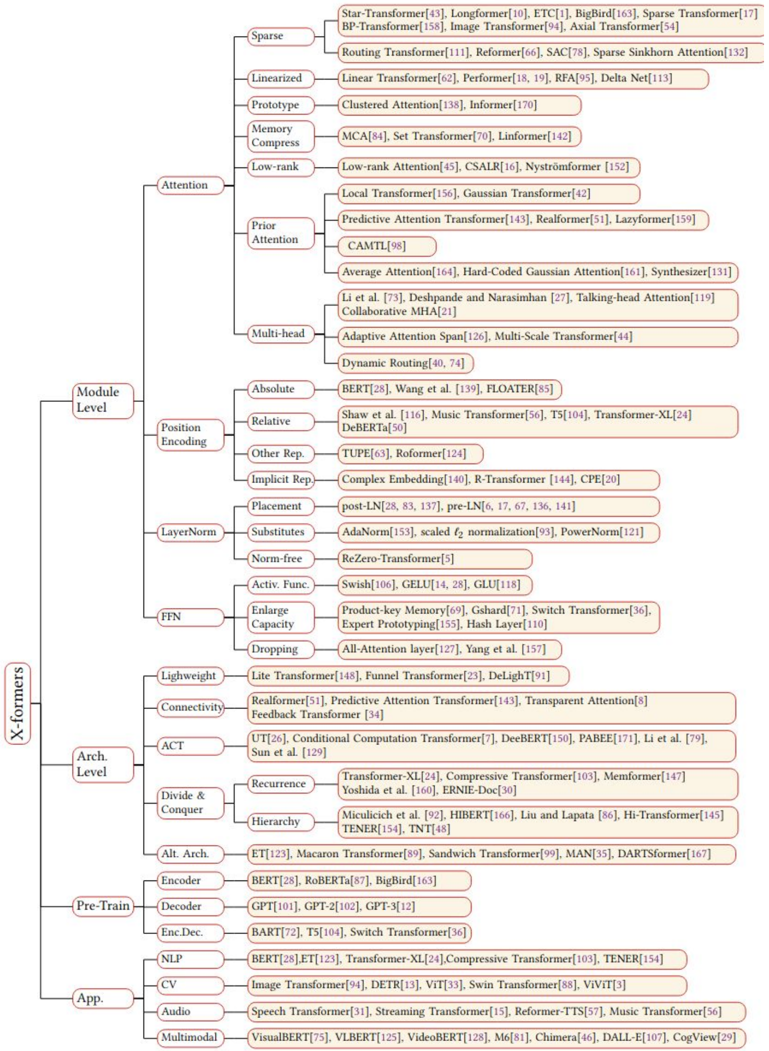
- Natural Language Processing
  - machine translation [35, 91 , 104 , 123 , 137]
  - language modeling [24, 103, 111 , 122]
  - named entity recognition [80 , 154]
  - pre-training on large-scale corpora
- Computer Vision
  - image classification [14 , 33 , 88]
  - object detection [13, 88, 168 , 172],
  - image generation [61, 94]
  - video processing [3, 115]
- Audio Applications
  - speech recognition [15, 31 , 41, 97],
  - speech synthesis [57, 76 , 169],
  - speech enhancement [65, 162]
  - and music generation [56]
- Multimodal Applications
  - visual question answering [55, 75, 77, 125],
  - visual common-sense reasoning [75, 125],
  - caption generation [22, 81, 128],
  - speech-to-text translation [46]
  - text-to-image generation [29, 81, 107].

Taxonomy of transformers

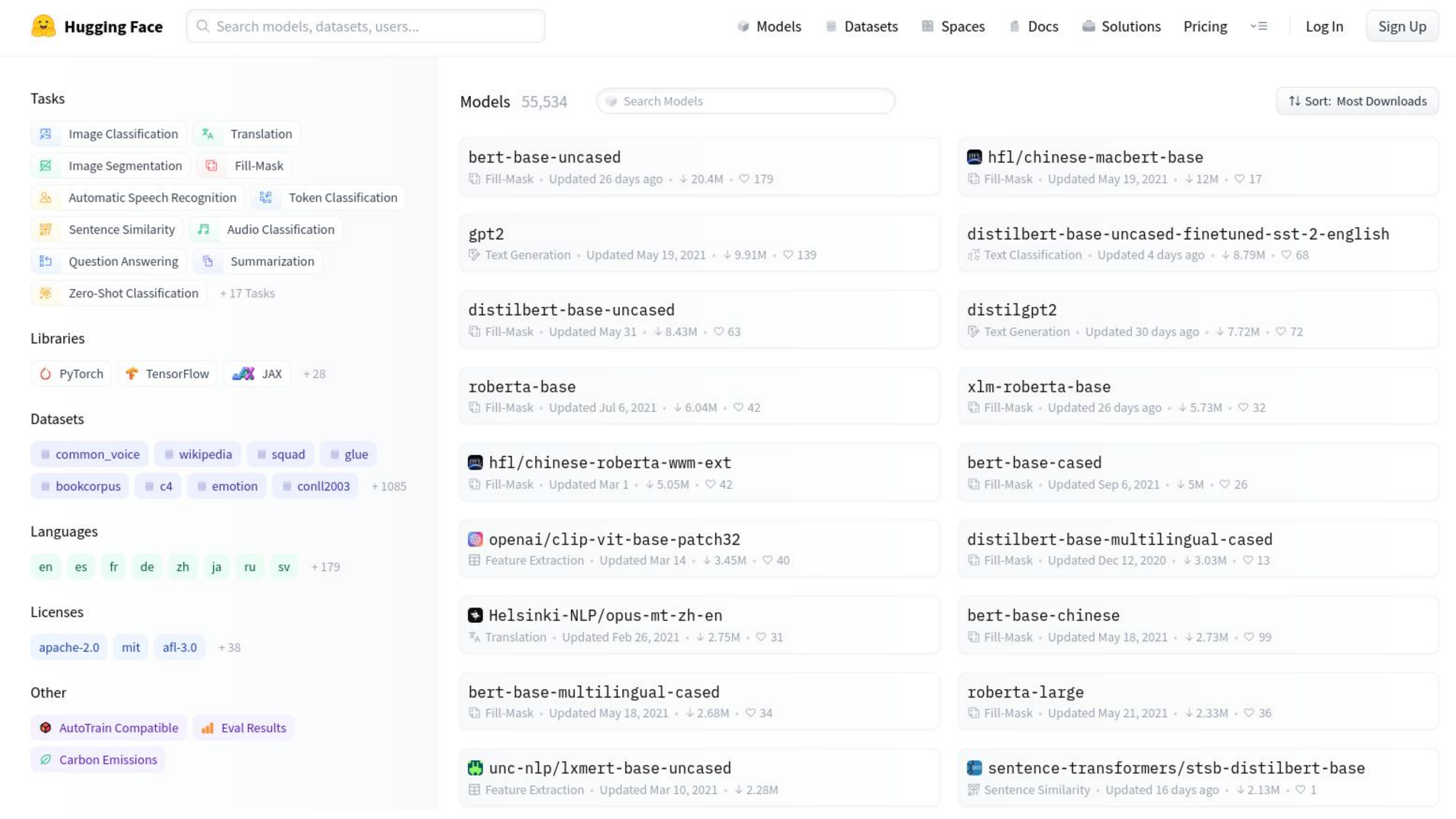


Lin et al. (2021)

Taxonomy of transformers



Lin et al. (2021)


Hugging Face
Search models, datasets, users...
Models
Datasets
Spaces
Docs
Solutions
Pricing
Log In
Sign Up
Tasks
Image Classification
Translation
Image Segmentation
Fill-Mask
Automatic Speech Recognition
Token Classification
Sentence Similarity
Audio Classification
Question Answering
Summarization
Zero-Shot Classification
+ 17 Tasks
Libraries
PyTorch
TensorFlow
JAX
+ 28
Datasets
common_voice
wikipedia
squad
glue
bookcorpus
c4
emotion
conll2003
+ 1085
Languages
en
es
fr
de
zh
ja
ru
sv
+ 179
Licenses
apache-2.0
mit
afl-3.0
+ 38
Other
AutoTrain Compatible
Eval Results
Carbon Emissions
Models 55,534
Search Models
Sort: Most Downloads
bert-base-uncased
Fill-Mask • Updated 26 days ago • 20.4M • 179
hfl/chinese-macbert-base
Fill-Mask • Updated May 19, 2021 • 12M • 17
gpt2
Text Generation • Updated May 19, 2021 • 9.91M • 139
distilbert-base-uncased-finetuned-sst-2-english
Text Classification • Updated 4 days ago • 8.79M • 68
distilbert-base-uncased
Fill-Mask • Updated May 31 • 8.43M • 63
distilgpt2
Text Generation • Updated 30 days ago • 7.72M • 72
roberta-base
Fill-Mask • Updated Jul 6, 2021 • 6.04M • 42
xlm-roberta-base
Fill-Mask • Updated 26 days ago • 5.73M • 32
hfl/chinese-roberta-wwm-ext
Fill-Mask • Updated Mar 1 • 5.05M • 42
bert-base-cased
Fill-Mask • Updated Sep 6, 2021 • 5M • 26
openai/clip-vit-base-patch32
Feature Extraction • Updated Mar 14 • 3.45M • 40
distilbert-base-multilingual-cased
Fill-Mask • Updated Dec 12, 2020 • 3.03M • 13
Helsinki-NLP/opus-mt-zh-en
Translation • Updated Feb 26, 2021 • 2.75M • 31
bert-base-chinese
Fill-Mask • Updated May 18, 2021 • 2.73M • 99
bert-base-multilingual-cased
Fill-Mask • Updated May 18, 2021 • 2.68M • 34
roberta-large
Fill-Mask • Updated May 21, 2021 • 2.33M • 36
unc-nlp/lxmert-base-uncased
Feature Extraction • Updated Mar 10, 2021 • 2.28M
sentence-transformers/stsb-distilbert-base
Sentence Similarity • Updated 16 days ago • 2.13M • 1

Transformers in NLP

GPT (Generative Pre-training Transformer)

- decoder-only transformer model
- pretrain with language modelling objective

$$L_1(T) = \sum_i \log P(t_i | t_{i-k}, \ldots, t_{i-1}; \theta)$$

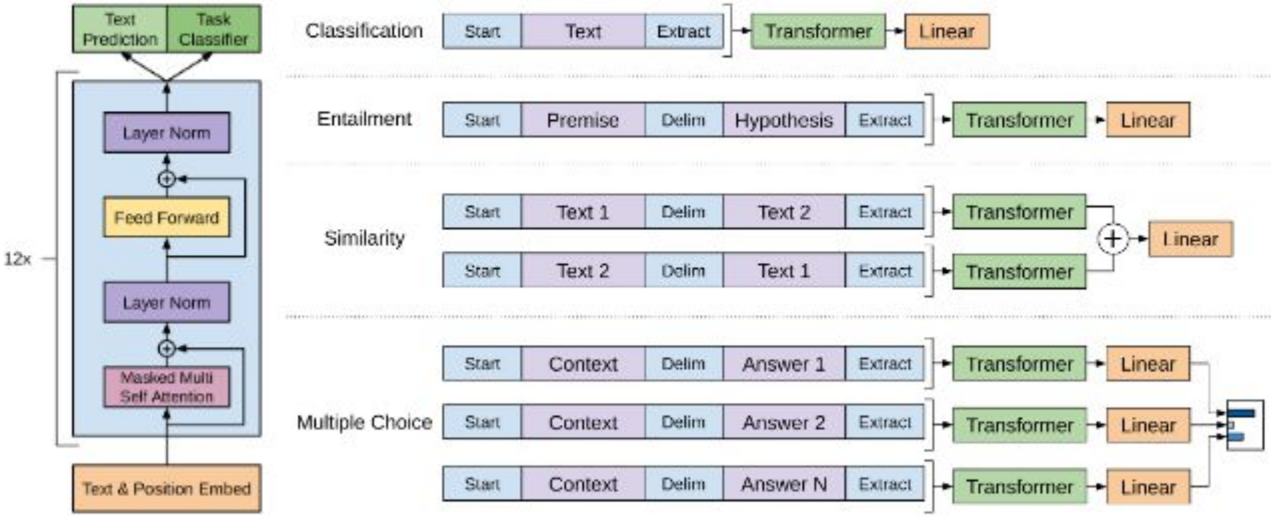
- transform input specifically for task
- finetune with task-specific objective

$$L_2(C) = \sum_{x,y} \log P(y | x_1, \ldots, x_n)$$

- zero-shot capabilities! (learning + task transfer)

GPT (Generative Pre-training Transformer)

- task-specific pipelines

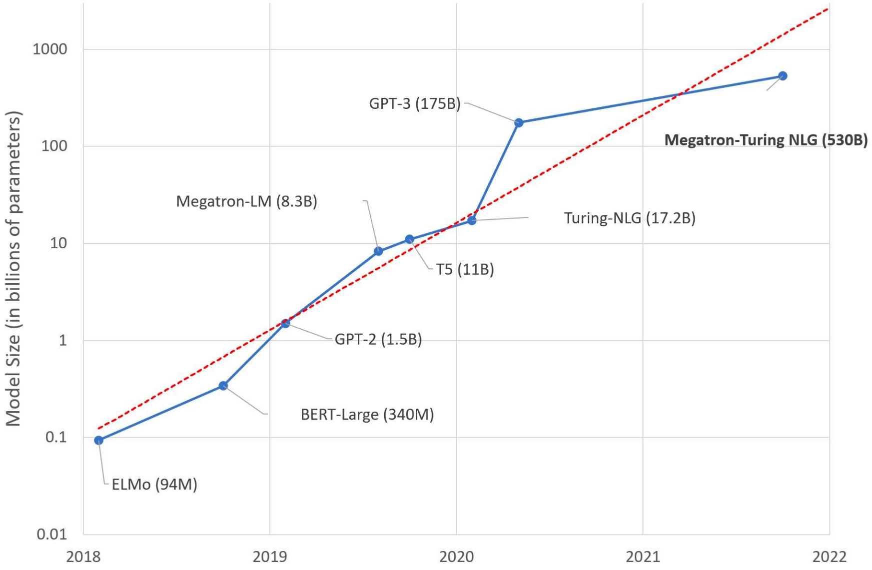


GPT (Generative Pre-training Transformer)

- GPT-1 - 117M params, 12 layers, 12 heads
  SOTA on 9 tasks

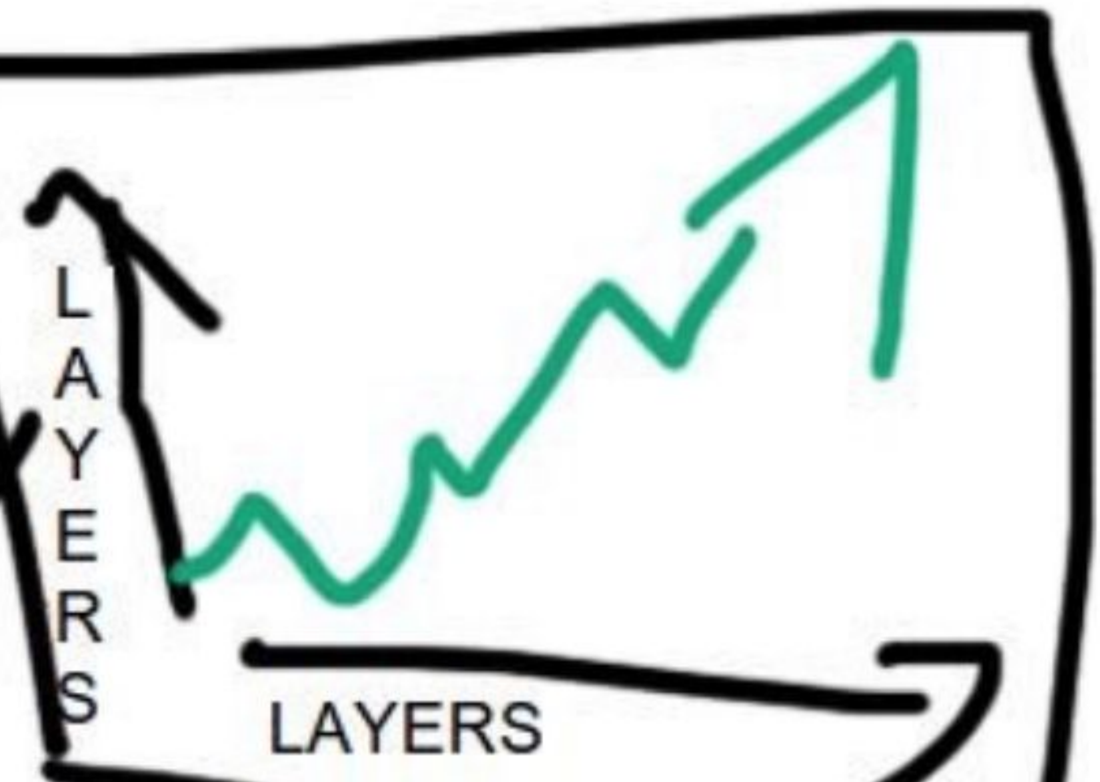
- GPT-2 - 1.5B params, 48 layers,
  SOTA on 7 LM (zero-shot),
  3 reading comprehension (zero-shot),
  translation (zero-shot)
  and LAMBADA tasks

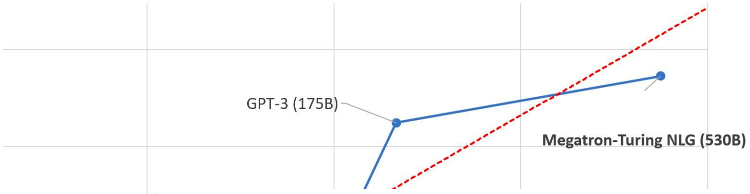
- GPT-3 - 175B params, 96 layers, 96 heads



AIRI × Университет Сириус

GPT (Generative Pre-training Transformer)





Back to the machine translation task

GPT: ...Never gonna give you up, never gonna...

Devlin et al., 2019

Back to the machine translation task

GPT: ...Never gonna give you up, never gonna...

	Never	gonna	give	you	up	,	never	gonna
Никогда	0,9	0,1						
тебя				1				
не	1							
подведу			0,5		0,5			
,						1		
никогда							0,9	0,1

Devlin et al., 2019

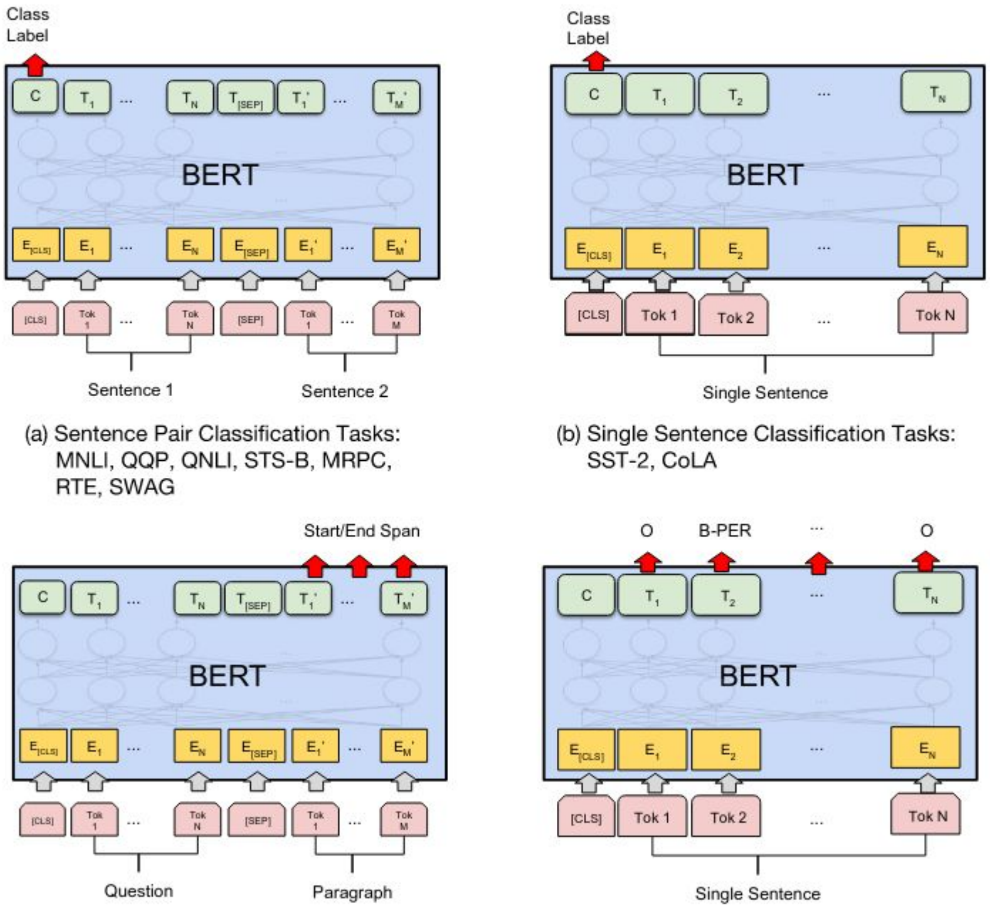
BERT (Bidirectional Encoder Representations using Transformer)

Recipe:

- encoder-only **bidirectional** transformer model
- uses BPE for text encoding
- adds sentence and positional encoding
- pretrain with MLM
  - corrupt 15% of tokens
  - replace 90% of corrupted tokens with *<mask>*
  - train model to retrieve them
- pretrain with NSP
  - what sequence comes next?
- finetune on a downstream task
- get SOTA on GLUE and SQuAD

Devlin et al., 2019

BERT



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

[Devlin et al., 2019]()

AIRI × Университет Сириус

GLUE

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**

Devlin et al., 2019

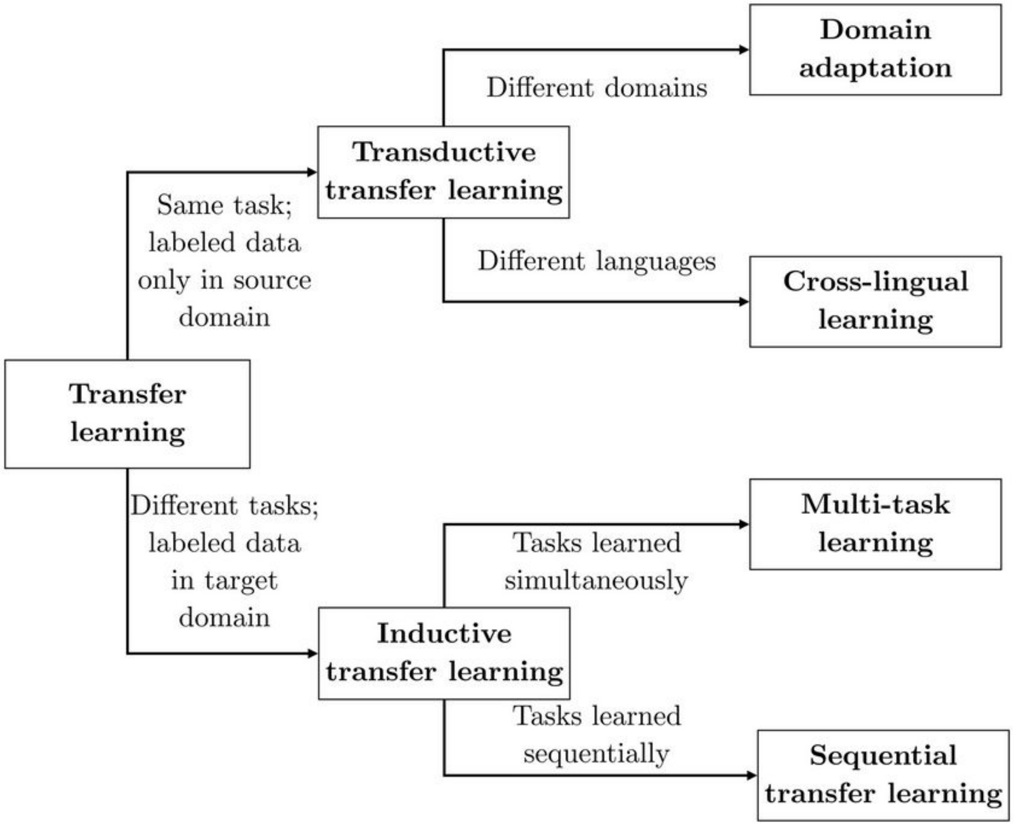
SQuAD

Devlin et al., 2019

RoBERTa (A Robustly Optimized BERT Pretraining Approach)

Idea: select hyperparameters and carefully train BERT

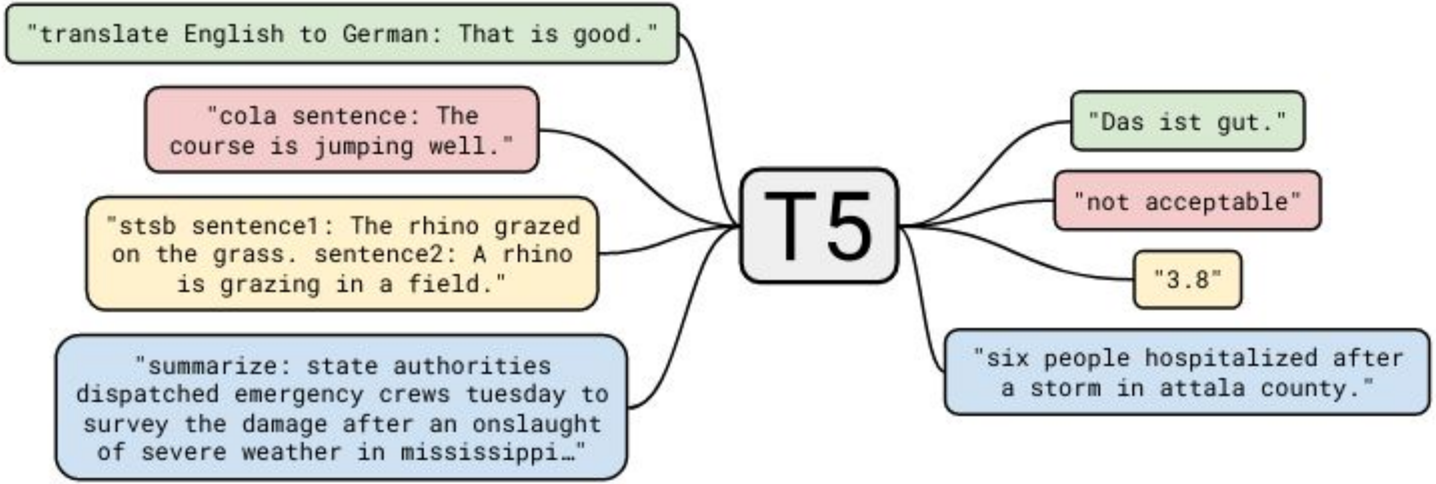
Transfer learning taxonomy



Ruder, 2019

T5 (Text-To-Text Transfer Transformer)

<u>Idea:</u> use multi-task learning for language tasks



T5

<u>Recipe:</u>

- create a shared text-to-text corpus (C4)
- take an enc-dec transformer from Vaswani et al.
- use relative positional encoding
- train the same model on summarization, question answering, text classification etc.
- finetune on GLUE, SuperGlue, SQuAD, ...
- Insights + Scale = State-of-the-Art

C4: The Colossal Clean Crawled Corpus

Authors state:

- We discarded any page with fewer than 5 sentences and only retained lines that contained at least 3 words.
- We removed any page that contained any word on the “List of Dirty, Naughty, Obscene or Otherwise Bad Words”
- Many of the scraped pages contained warnings stating that Javascript should be enabled so we removed any line with the word Javascript.
- Some pages had placeholder “lorem ipsum” text; we removed any page where the phrase “lorem ipsum” appeared.
- Some pages inadvertently contained code. Since the curly bracket “{” appears in many programming languages (such as Javascript, widely used on the web) but not in natural text, we removed any pages that contained a curly bracket.
- To deduplicate the data set, we discarded all but one of any three-sentence span occurring more than once in the data set

Pretraining objectives

Objective	Inputs	Targets
Prefix language modeling	Thank you for inviting	me to your party last week .
BERT-style Devlin et al. (2018)	Thank you <M> <M> me to your party apple week .	*(original text)*
Deshuffling	party me for your to . last fun you inviting week Thank	*(original text)*
MASS-style Song et al. (2019)	Thank you <M> <M> me to your party <M> week .	*(original text)*
I.i.d. noise, replace spans	Thank you <X> me to your party <Y> week .	<X> for inviting <Y> last <Z>
I.i.d. noise, drop tokens	Thank you me to your party week .	for inviting last
Random spans	Thank you <X> to <Y> week .	<X> for inviting me <Y> your party last <Z>

Objective	GLUE	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
Prefix language modeling	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
BERT-style (Devlin et al., 2018)	**82.96**	**19.17**	**80.65**	**69.85**	**26.78**	**40.03**	**27.41**
Deshuffling	73.17	18.59	67.61	58.47	26.11	39.30	25.62

Limitations

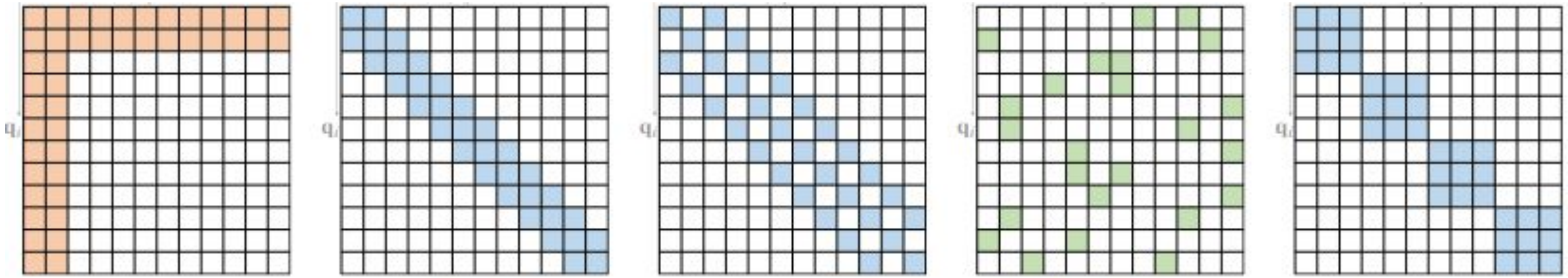
What are the problems with Transformer

1) Complexity.
   The complexity of self-attention is O (N*N), where N is the input size.
   The attention module becomes a bottleneck when dealing with long sequences.

2) Structural prior.
   Self-attention does no assume any structural bias over inputs. Even the order information is also needed to be learned from training data. Therefore, Transformer is usually easy to overfit on small or moderate-size data

Sparse attention

Limit the attention mask to reduce computational complexity



(a) global (b) band (c) dilated (d) random (e) block local

Artificial Intelligence
Research Institute

airi.net

airi_research_institute

AIRI Institute

AIRI Institute

AIRI_inst

artificial-intelligence-research-institute

Telegram

Github of NLP Course

Feedback