



UiT The Arctic
University of Norway




BIO-AI LAB | ARCTIC LLM WORKSHOP 2023
Large Language Models

Day 1 - Session 1

Introduction to LLM Technology

Himanshu Buckchash

 himanshu.buckchash@uit.no

27. Oct 2023



give the month name that has six sundays



To determine the month that has six Sundays, we need to find a month that has 31 days and starts on a Sunday.

There are only seven possible starting days for a month, and each of them occurs 52 times in a non-leap year. This means that there are 365 days in a year, which is equal to 7 days per week times 52 weeks plus one day.

Therefore, the only month that can have six Sundays is a month that starts on a Sunday and has 31 days.

After checking the calendar, we can see that the only month that meets these criteria is **July**.

?



multiply 324 with 234



To multiply 324 with 234, we can use the standard multiplication algorithm as follows:

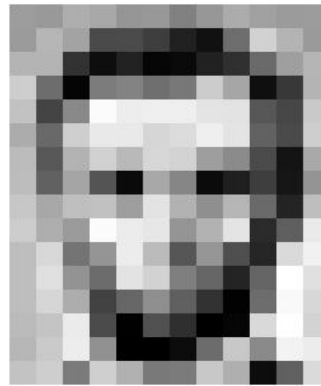
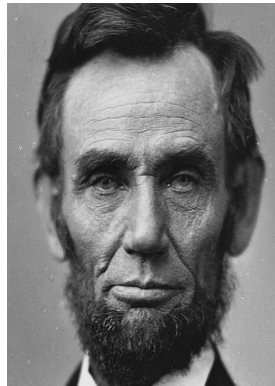
Copy code

```
324
x 234
-----
 648   (4 x 324)
+2592  (3 x 324 with a zero added)
+6480  (2 x 324 with two zeros added)
-----
75696
```

Therefore, 324 multiplied by 234 is equal to 75,696.

Answer: 75816

Preliminaries: Representation — Numbers



187	163	174	168	150	162	129	161	172	161	165	166
155	182	163	74	75	62	93	17	116	210	180	154
180	180	50	14	34	6	10	93	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	35	101	255	224
190	214	173	66	103	143	96	90	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

187	163	174	168	150	162	129	161	172	161	165	166
155	182	163	74	75	62	93	17	116	210	180	154
180	180	50	14	34	6	10	93	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	35	101	255	224
190	214	173	66	103	143	96	90	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

You like this talk



You
like
this
talk

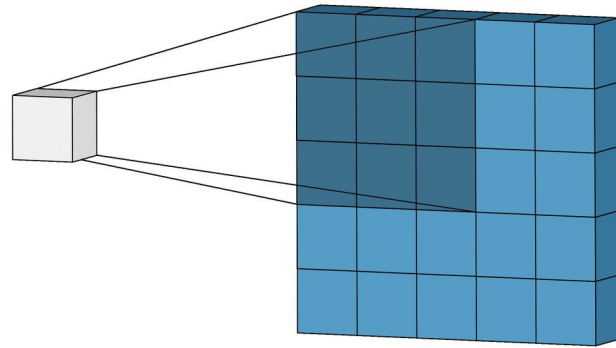
a vector



.031, .284, ..., .701
.972, .061, ..., .548
.019, .805, ..., .732
.979, .294, ..., .101

a matrix

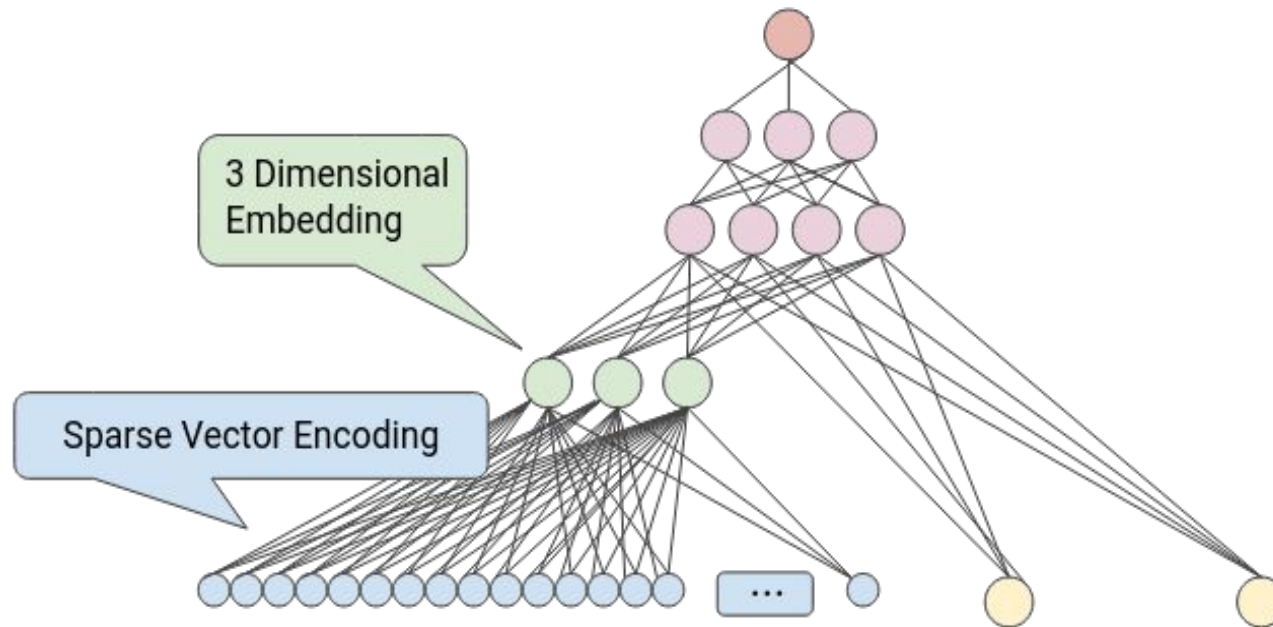
Preliminaries: Convolution



Convolution

Repeated matrix multiplication

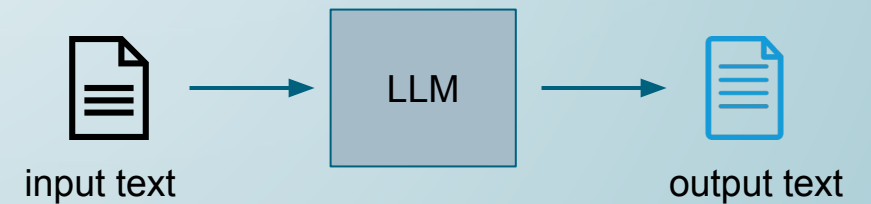
Preliminaries: Embedding



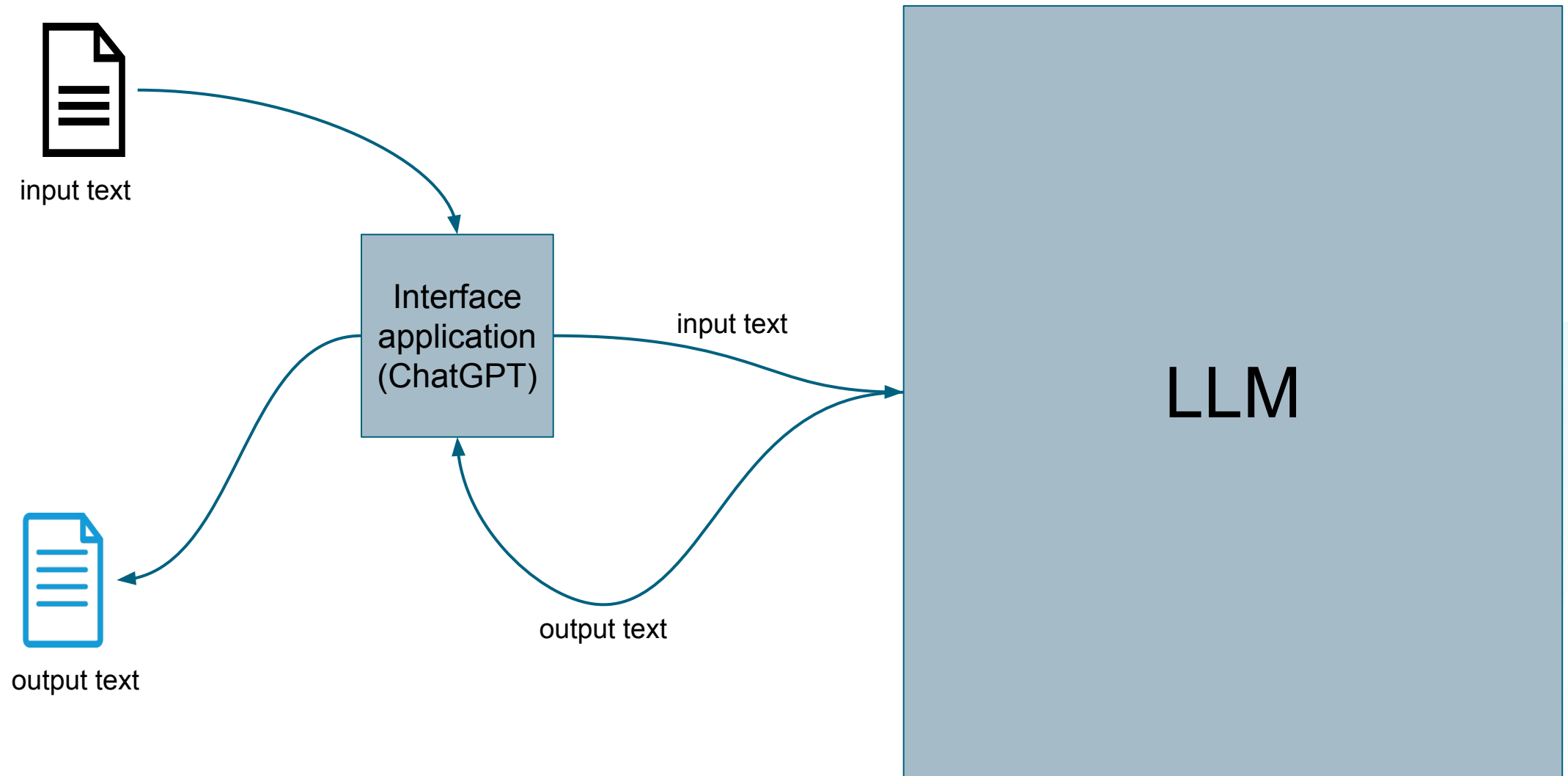
Multi stepped (**chained**) matrix multiplication
Compression

SECTION 1

Journey of text through LLM



Journey of text through LLM



Task



Can U expand LLM? $\xrightarrow{\text{LLM}}$ Large Language Modeling

LLM Input preparation



Can U expand LLM?



can u expand llm?

Normalization

Remove accent, spaces,
lowercasing

LLM Input preparation



Can U expand LLM?



can u expand llm?



{can,u,expand,llm,?}

Normalization

Remove accent, spaces, lowercasing

Pre-tokenization

Break at whitespace and punctuation

LLM Input preparation



Can U expand LLM?



Normalization

Remove accent, spaces, lowercasing

can u expand llm?



Pre-tokenization

Break at whitespace and punctuation

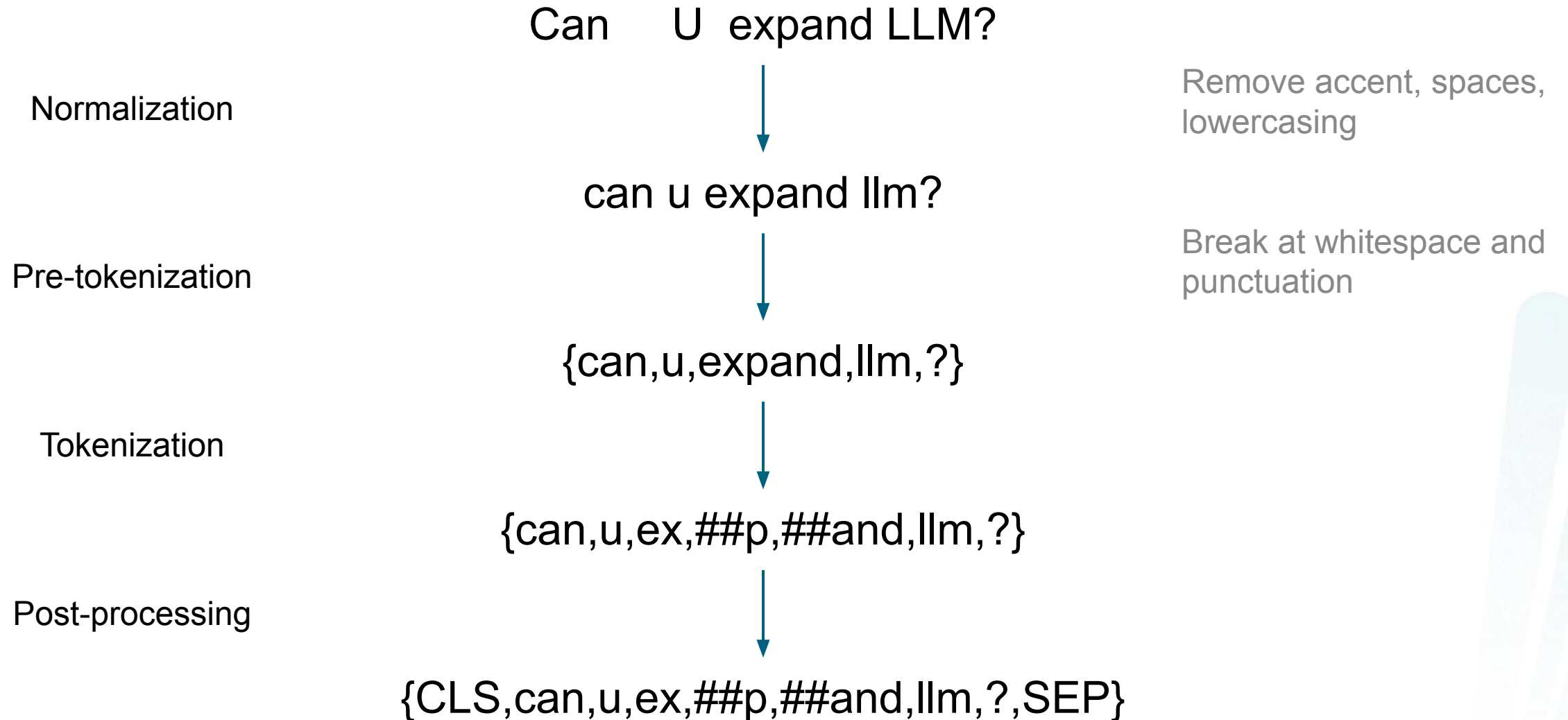
{can,u,expand,llm,?}



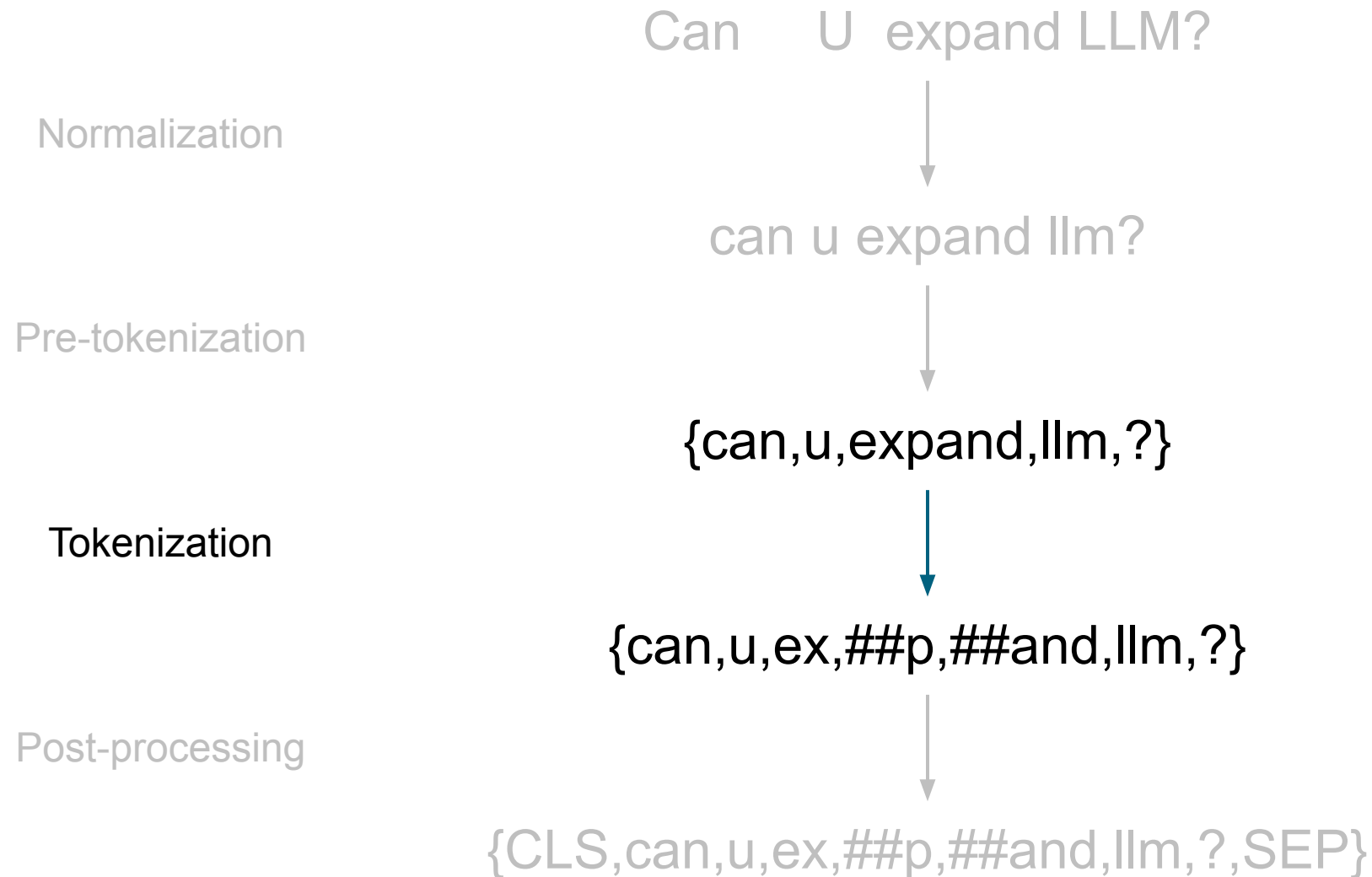
Tokenization

{can,u,ex,##p,##and,llm,?}

LLM Input preparation



Tokenization: Vocabulary creation



Tokenization: Vocabulary creation



Corpus of text → Tokenization algorithm → Vocabulary

Corpus of text

C
o
r
p
u
s
o
f
t
e
x
t

C
o
r
p
u
s
o
f
t
e
x
t
x
t

C
o
r
p
u
s
o
f
t
e
x
t
x
t
e
x
t

Tokenization algorithms



Byte pair encoding

WordPiece

SentencePiece

Unigram

Embedding



{CLS,can,u,ex,##p,##and,llm,?,SEP}



{000,.648,.61,.088,.094,.761,.032,.076,.809}

Embedding: fixed length floats



CLS		000, 000
can		.648, .032
u		.619, .088
ex		.094, .761
##p	→	.032, .076
##and		.809, .094
llm		.093, .461
?		.038, .076
SEP		-inf, -inf

Static vs. **Dynamic**

Causal Language Modeling (CLM)

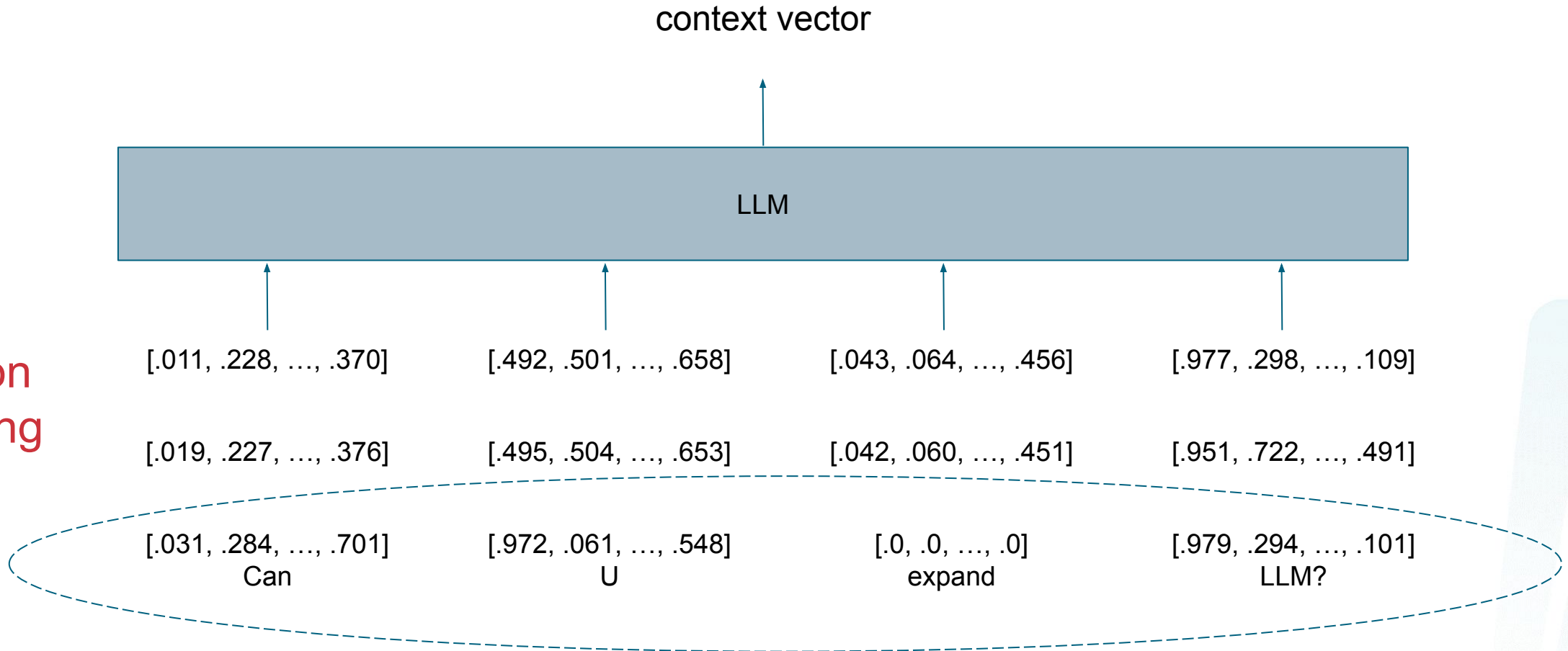
Autoregressive Language Inference



Autoregressive Generation: Causal Language Modeling (CLM)



Position encoding

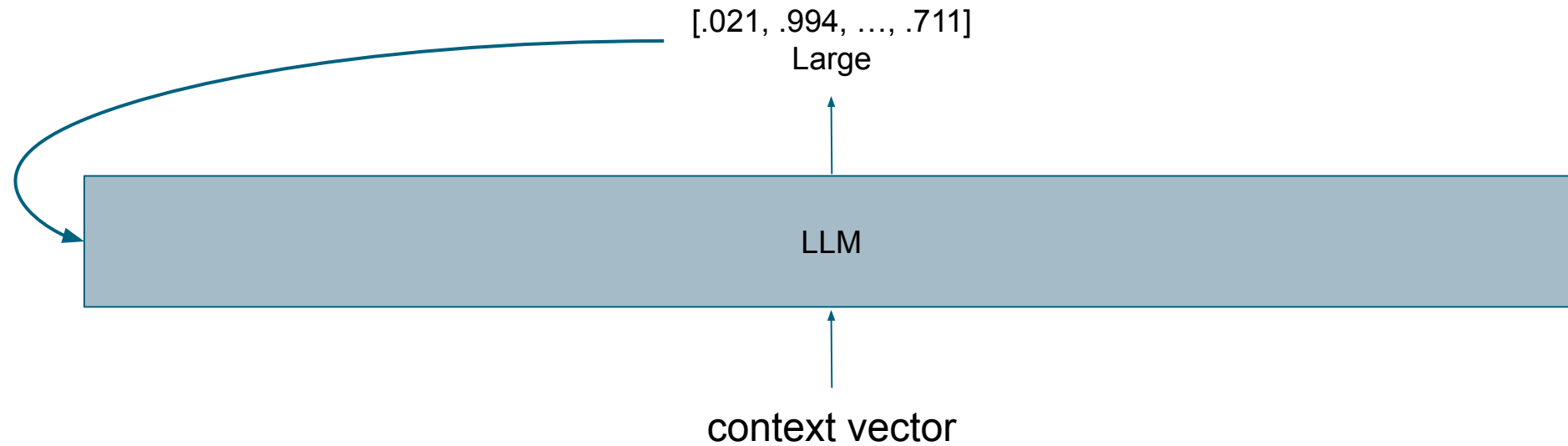


Context length, Block length, Chunk length

Autoregressive Generation: Causal Language Modeling (CLM)



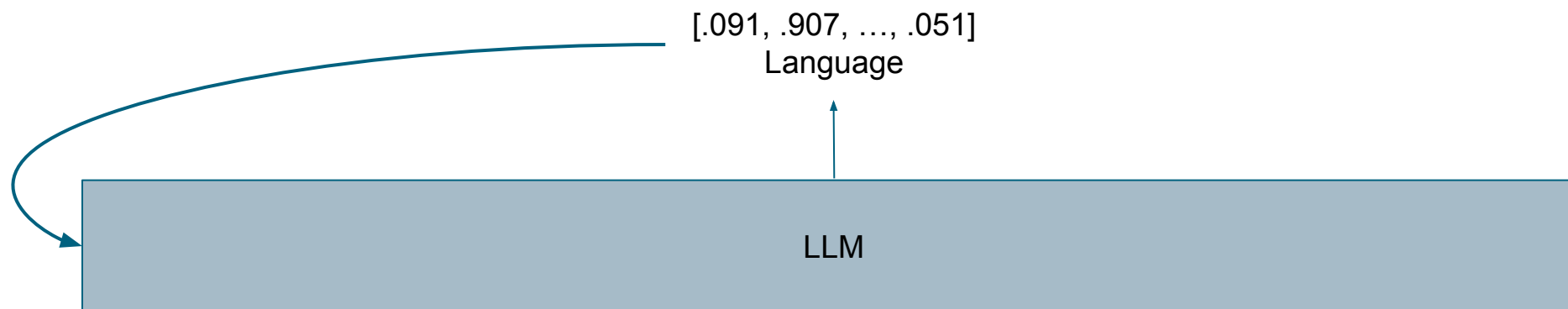
Output:



Autoregressive Generation: Causal Language Modeling (CLM)



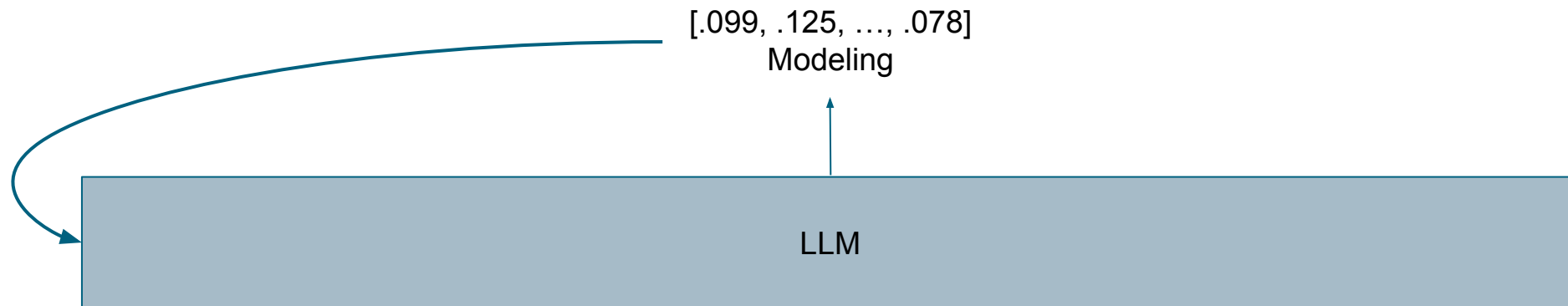
Output: **Large**



Autoregressive Generation: Causal Language Modeling (CLM)



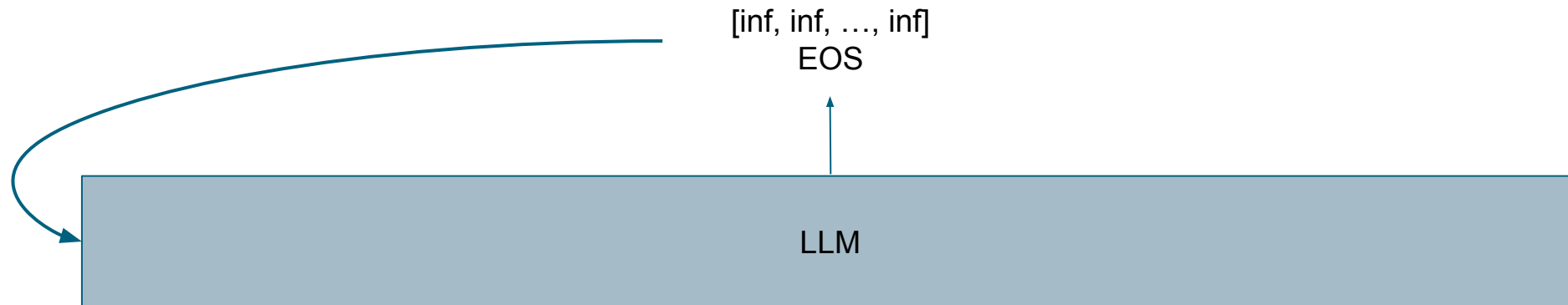
Output: **Large Language**



Autoregressive Generation: Causal Language Modeling (CLM)



Output: Large Language Modeling



Autoregressive Generation: **Decoding Stack**



[.021, .994, ..., .711]

Large

[.091, .907, ..., .051]

Language

[.099, .125, ..., .078]

Modeling

[inf, inf, ..., inf]

SEP

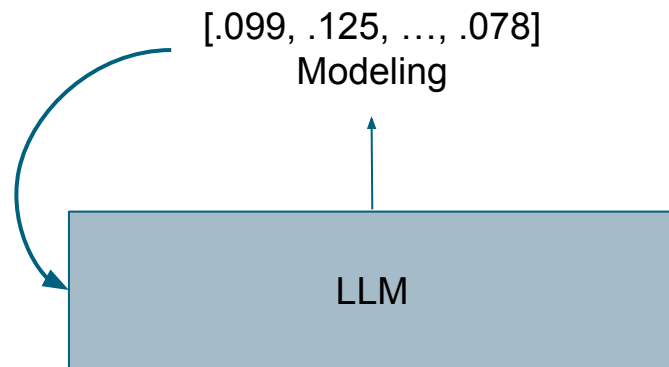


Large Language Modeling

Complex Probability Distribution: Greedy Decoding



Output: Large Language

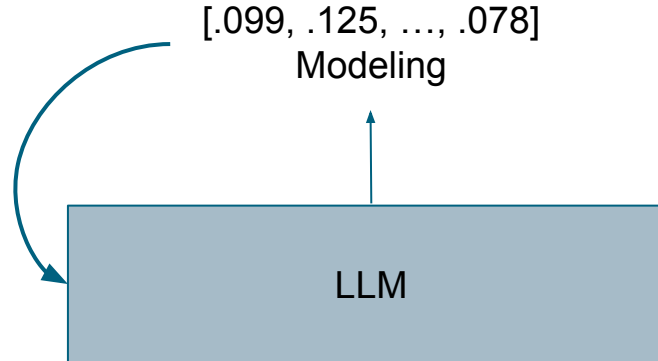


Large	.0002
Language	.0003
Modeling	.0059

Beam Search: Non greedy approach



Output: Large Language



Search Frontier: {Language, Modeling}

Controlling creativity: **Temperature**



High temperature: more creative generation

Cat sits on: **moon, mars, sea**

Low temperature: more realistic generation

Cat sits on: **sofa, floor**

Beam Search: **Top-P**



.0859
.0602
.0548
.0248
.0088
.0018
.0012
.0008
.0001

Top **P** % tokens are considered for generation

LLM architectures

LLM architectures



Encoder only

BERT

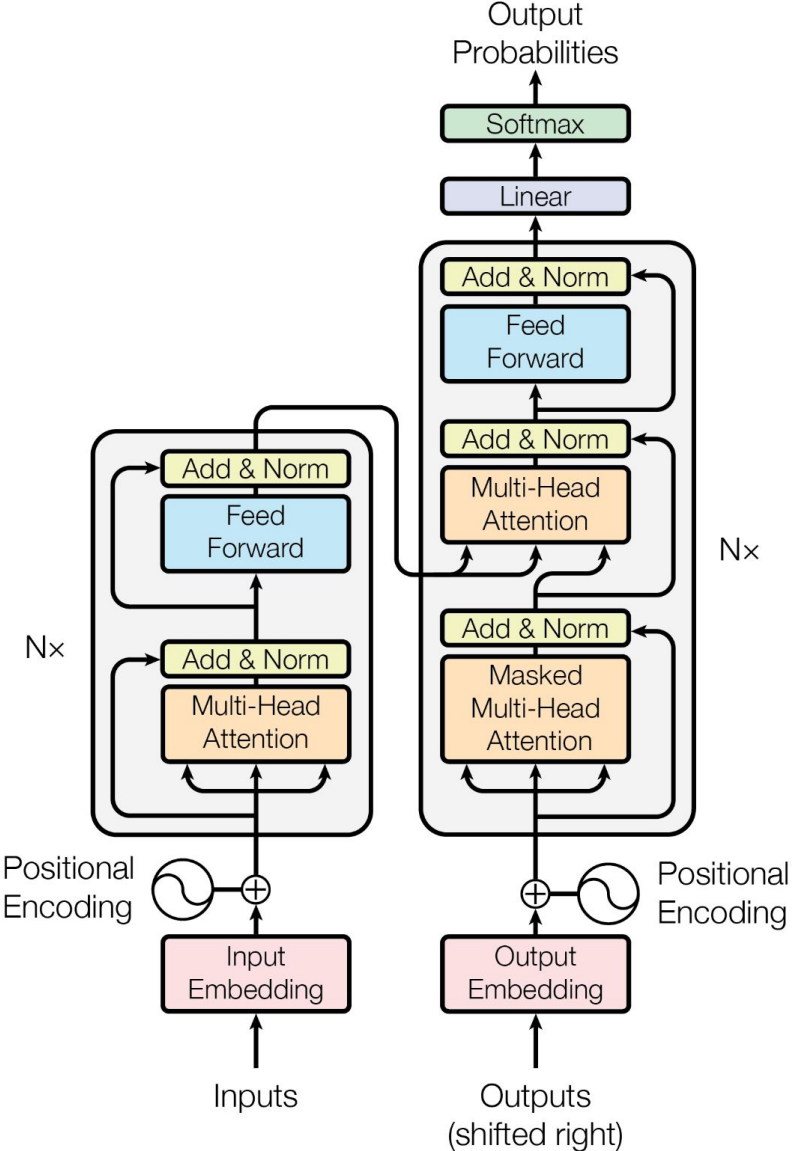
Encoder
decoder

GPT - 2

Decoder only

GPT - 3,4

encoder - decoder



Vaswani *et al.*, 2017, Attention is all you need

LLM modelling (tasks) types



Masked
Language
Modeling
(MLM)

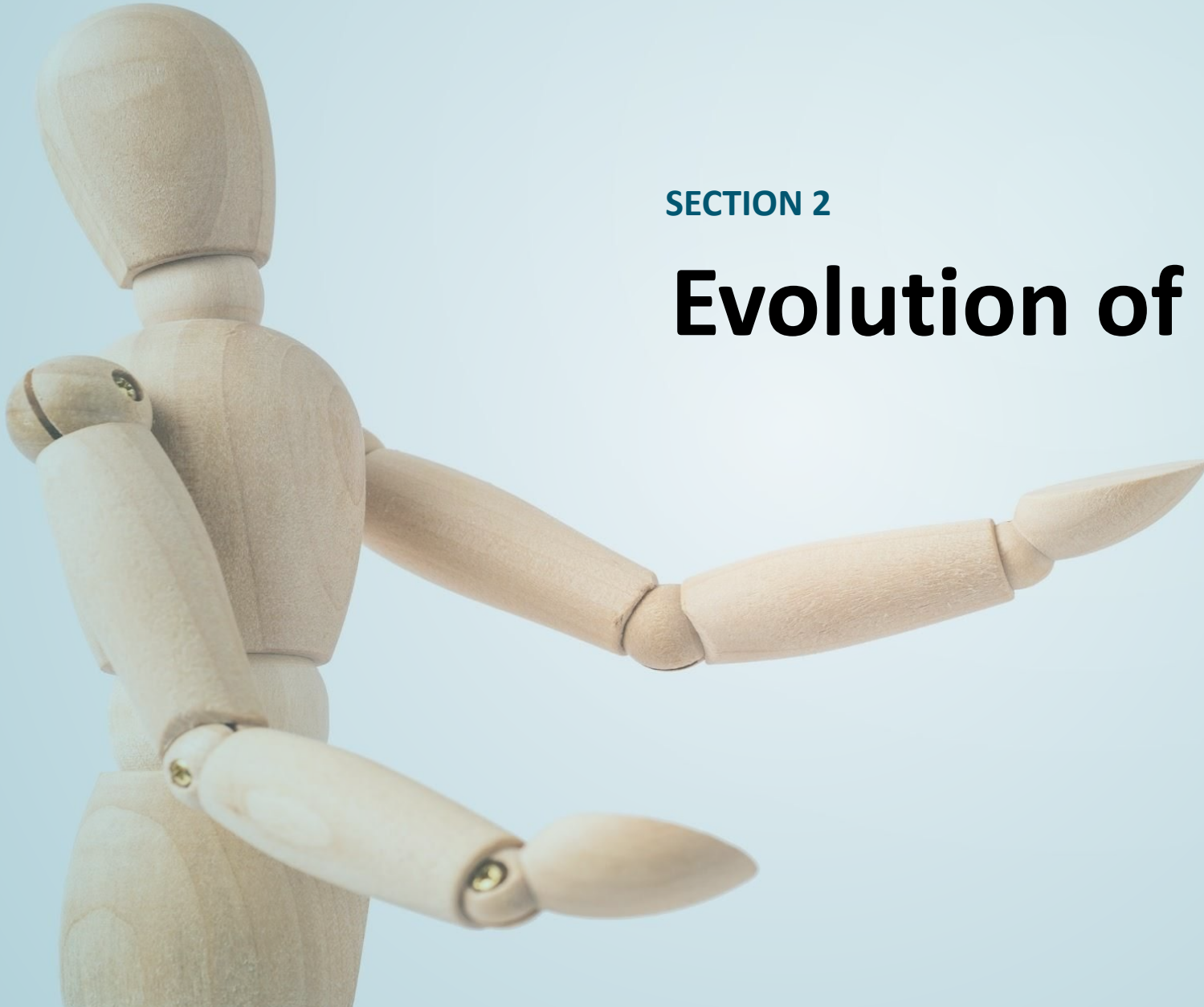
BERT

Causal
Language
Modeling
(CLM)

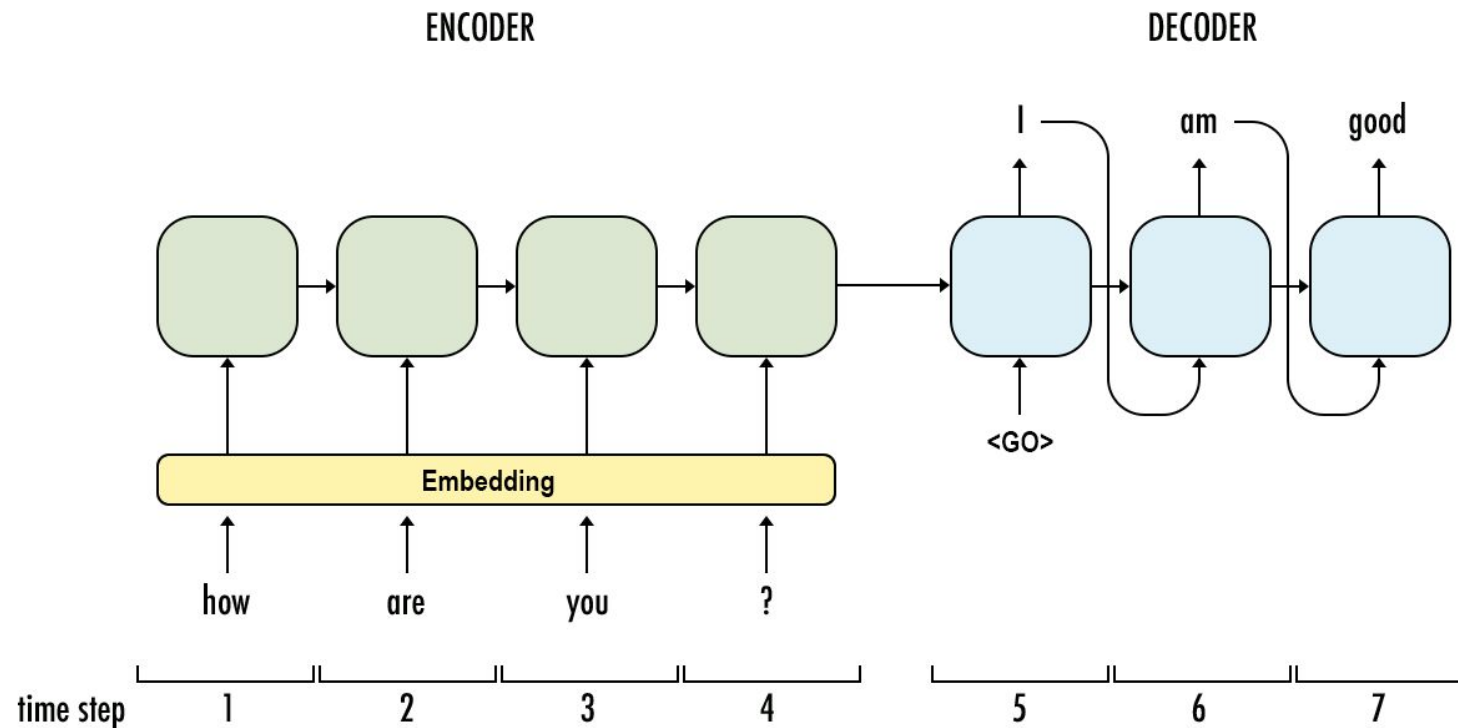
GPT - 2,3,4

SECTION 2

Evolution of LLM

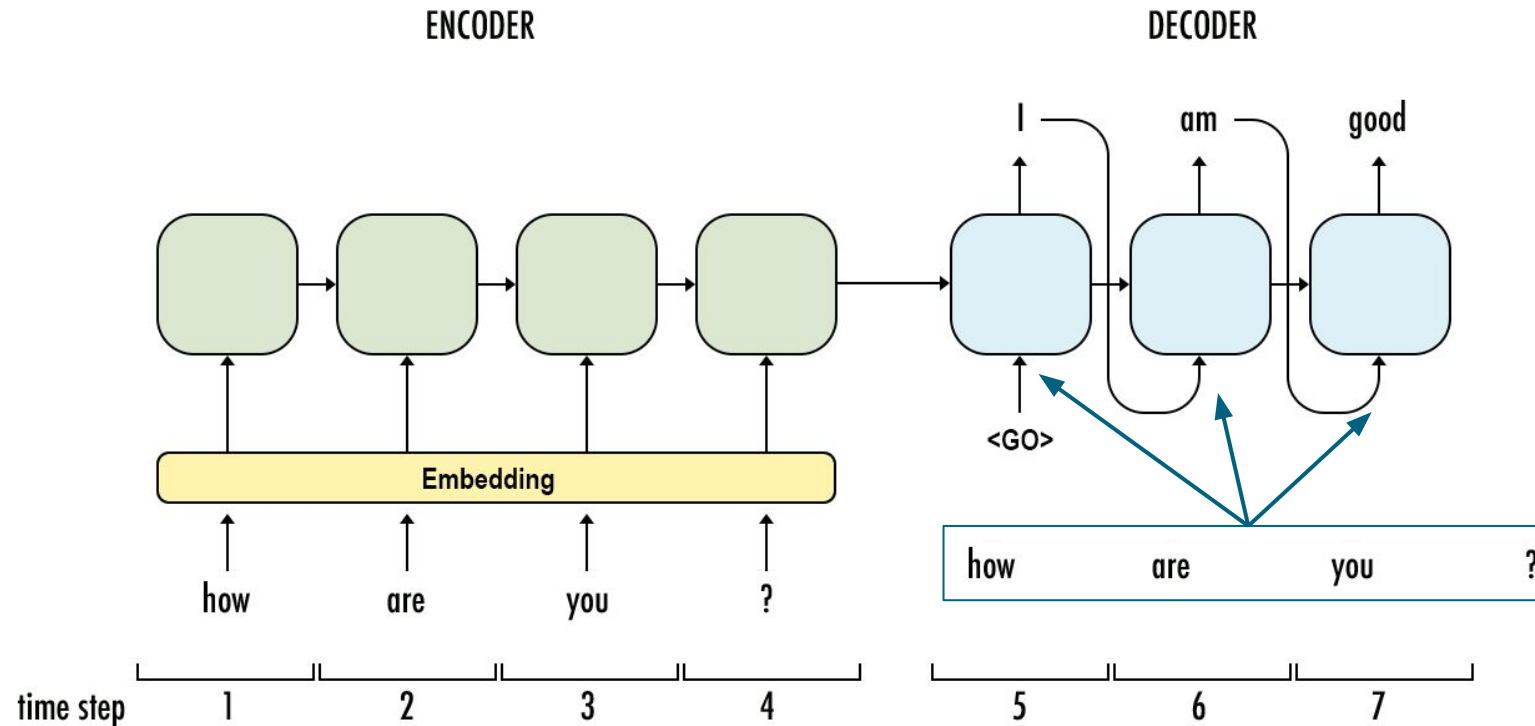


Sequence to Sequence Models



Recurrent neural networks (**without** attention)
Problem: **short context**

Sequence to Sequence Models (**with attention**)



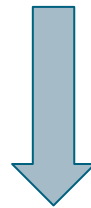
Recurrent neural networks (**with attention**)

Problem: **vanishing gradients**

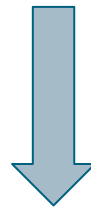
Path Breaking Idea (**Transformer**)



RNN (attention) → **Attention(RNN)**

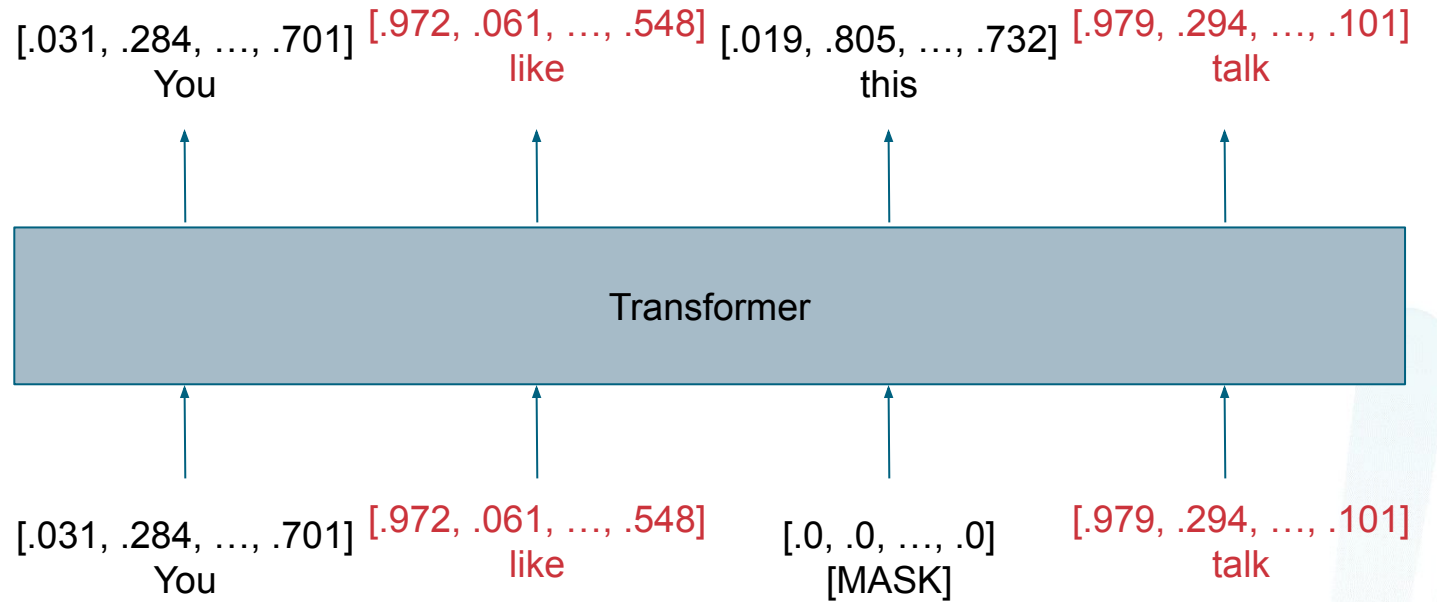
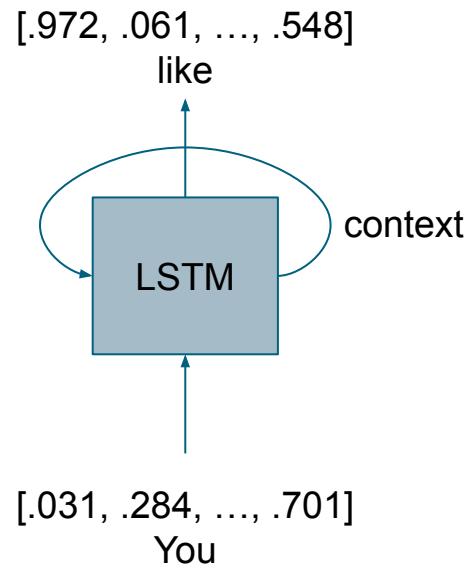


Sequential learning → **Parallel Learning**

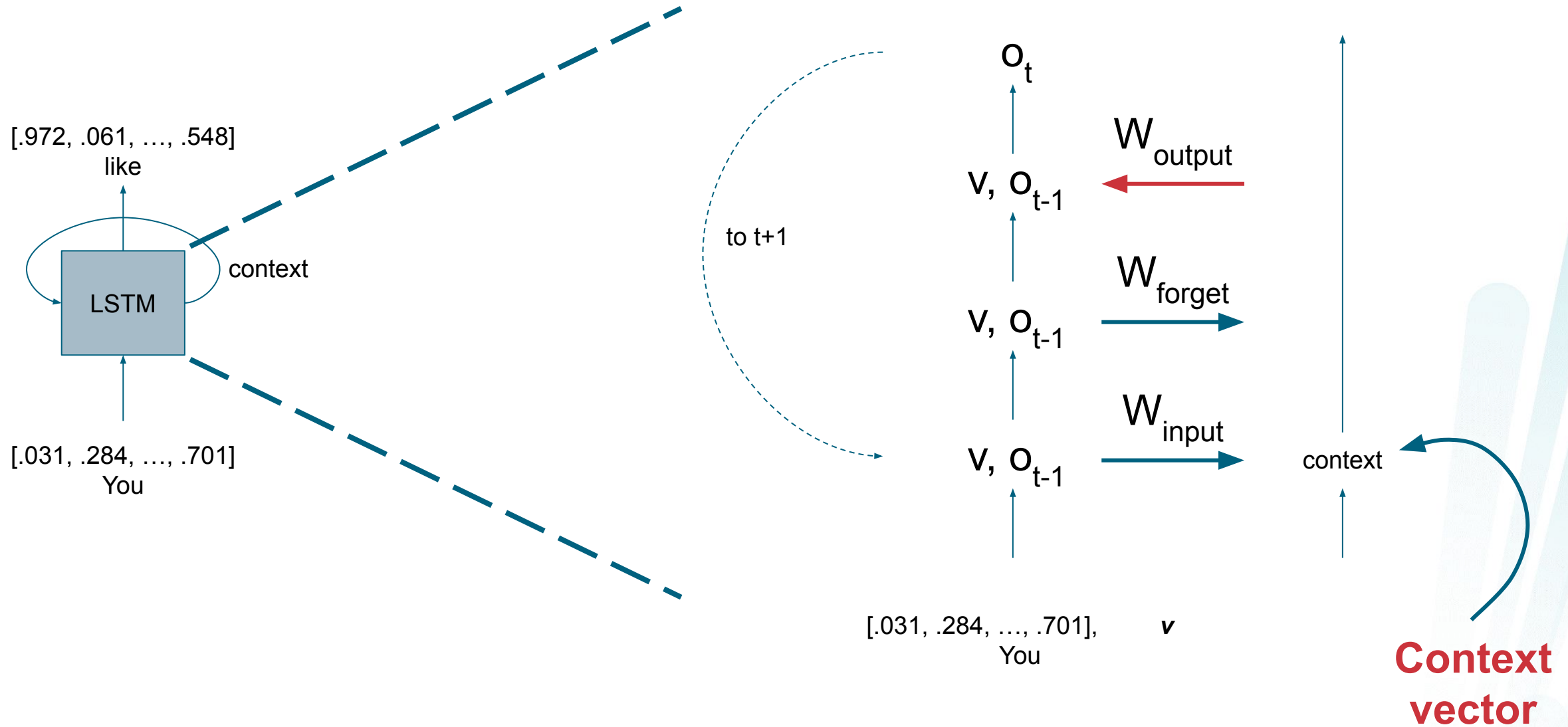


RNN → **Transformer**

RNN vs. Transformer



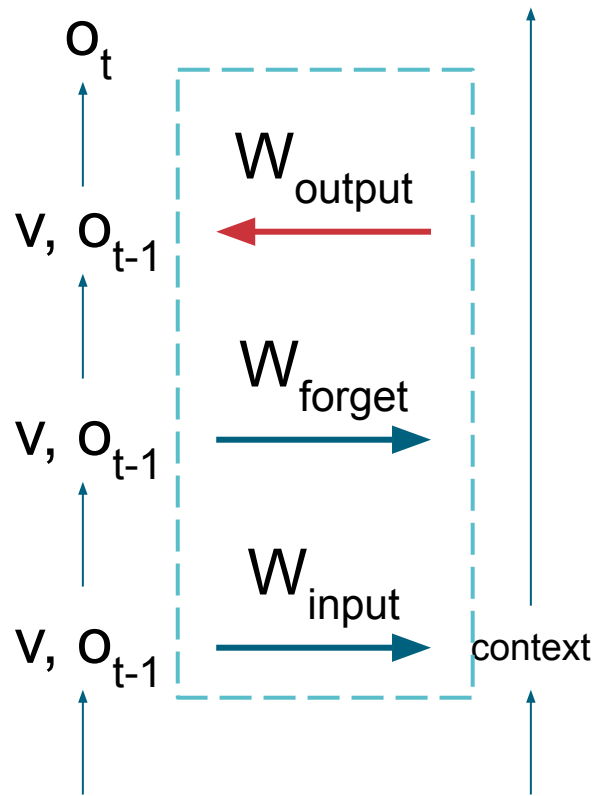
LLM evolution: Recurrent Neural Networks (RNNs)



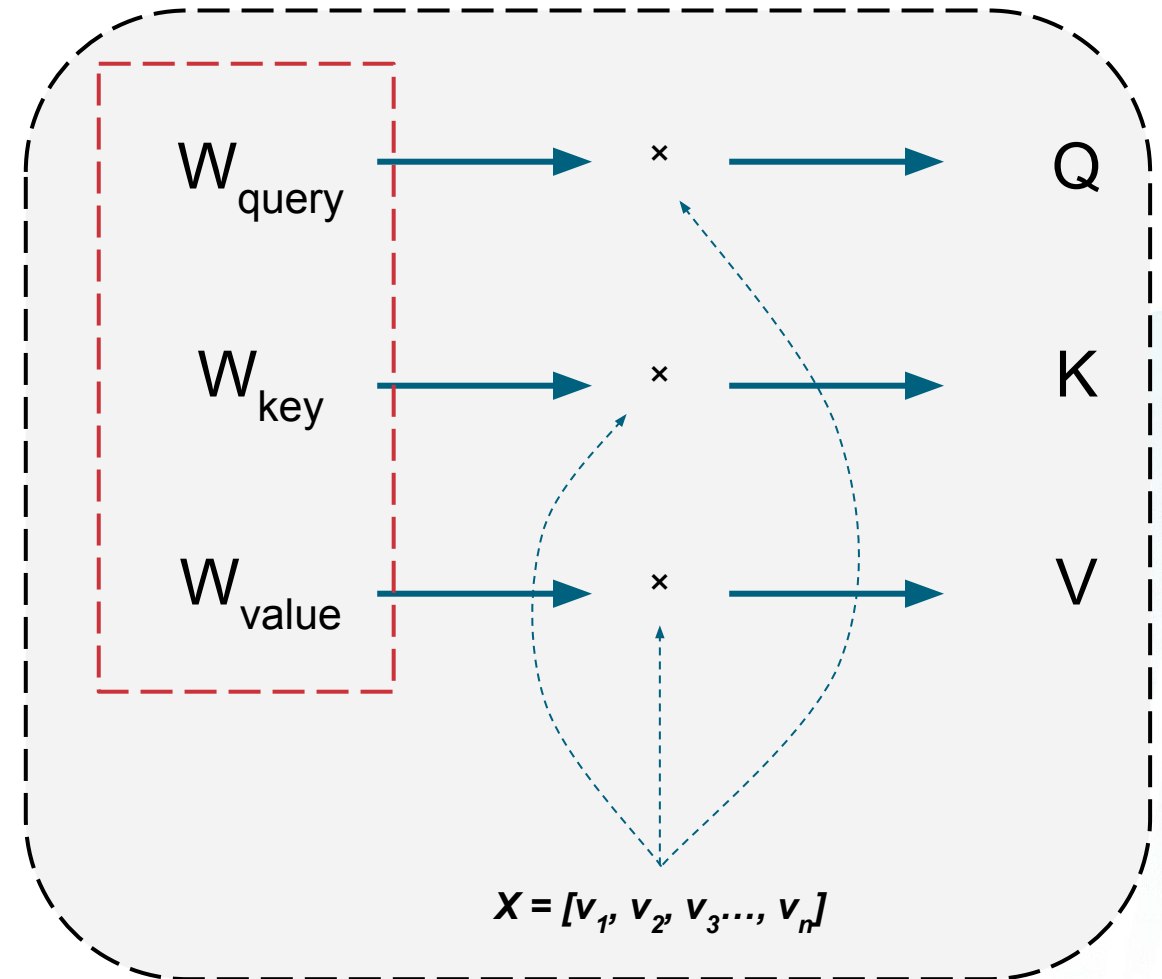
LLM evolution: Context/Ops Symmetry (RNN – Transformer)



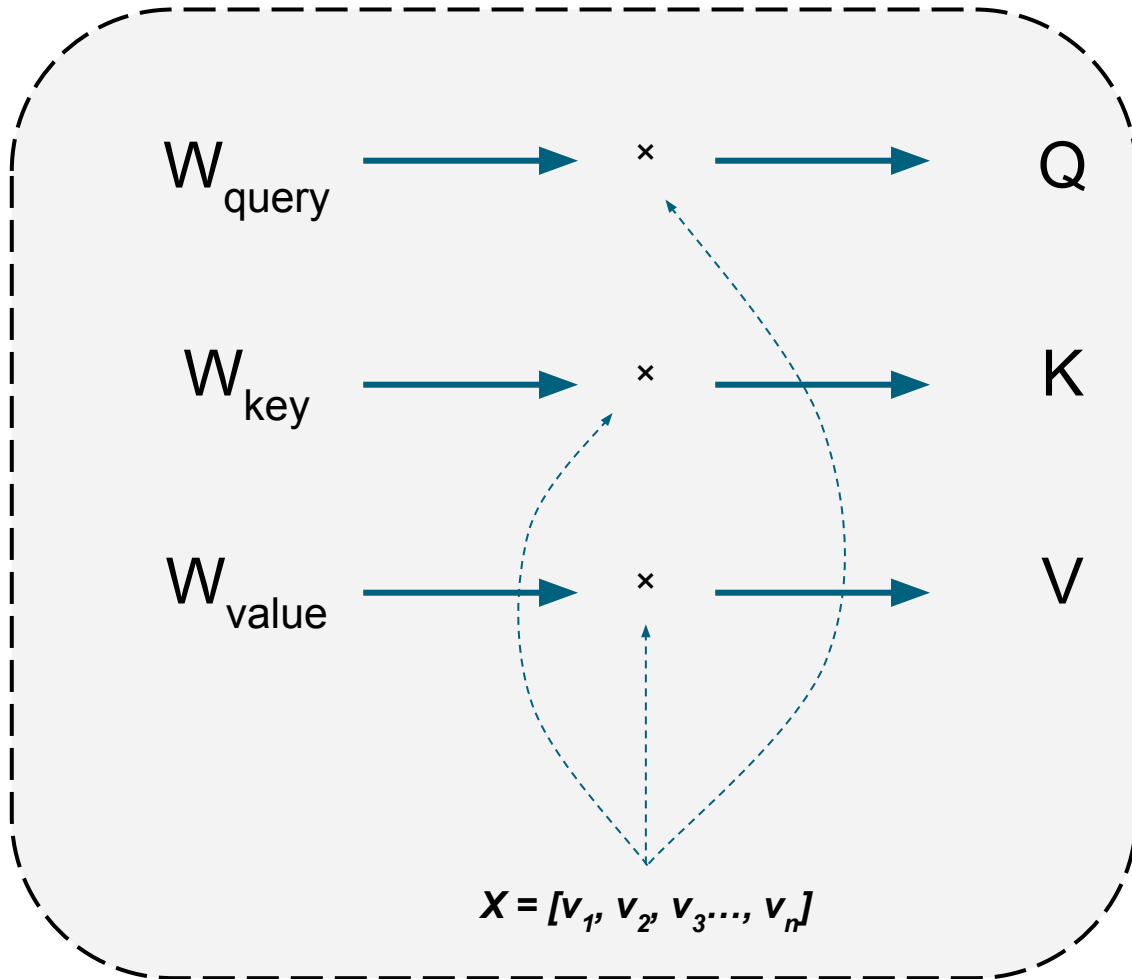
Self attention (heart of transformer)



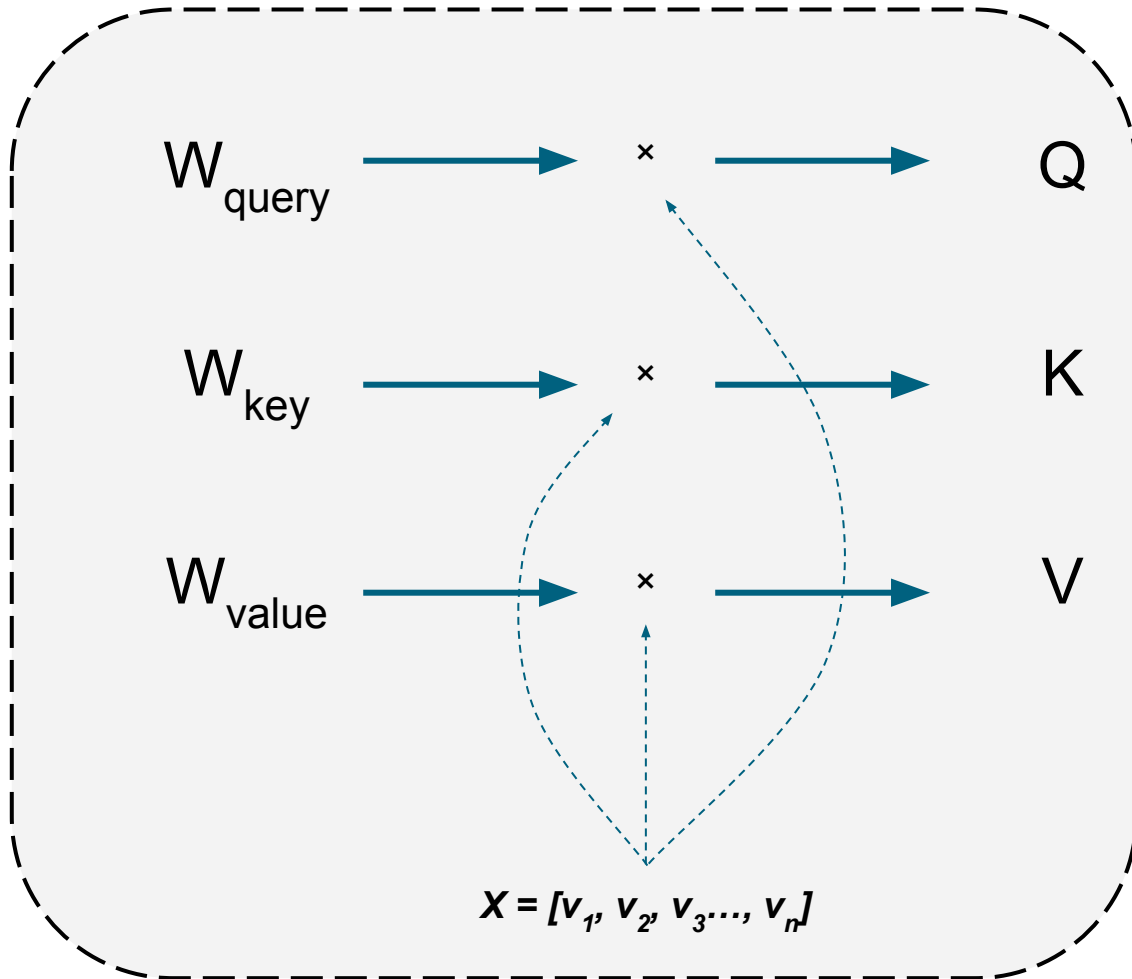
[.031, .284, ..., .701], v
You



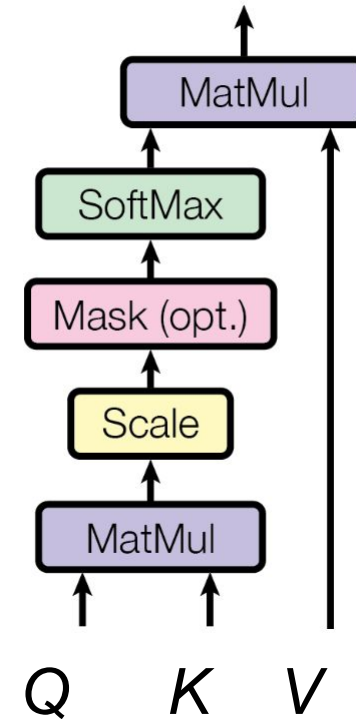
LLM evolution: Transformer (**self-attention**)



LLM evolution: Transformer (**self-attention**)



Attention(Q, K, V)

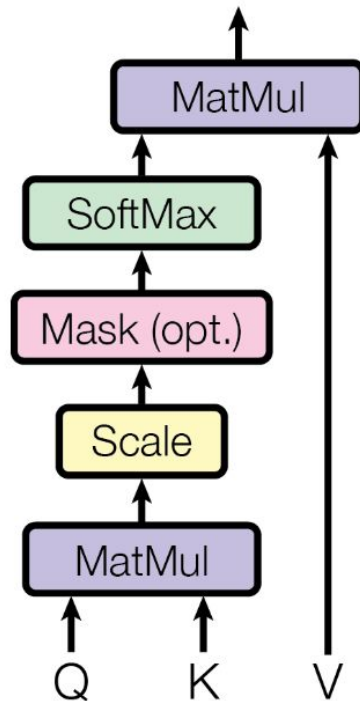


$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

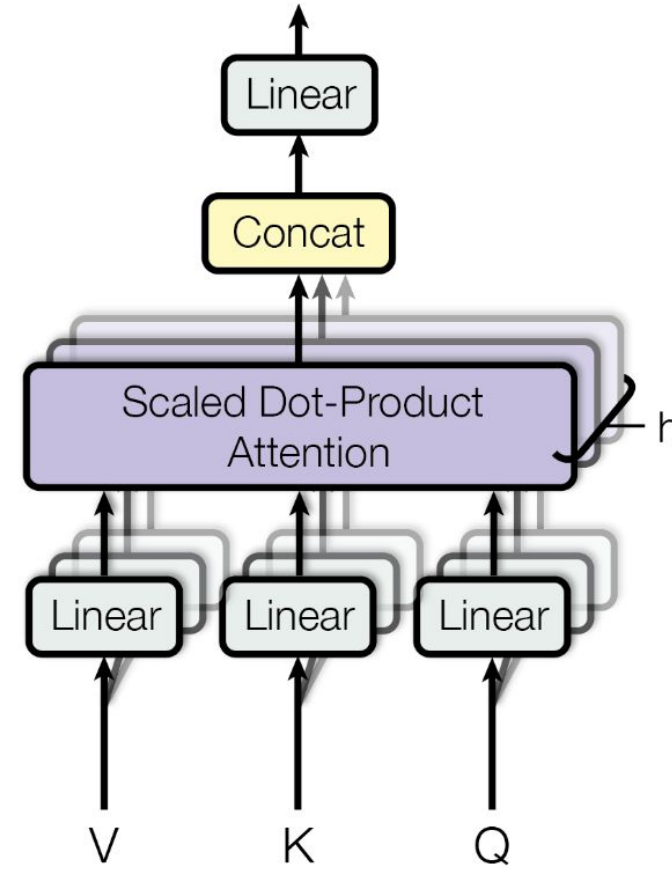
LLM evolution: Transformer (**multi-head attention**)



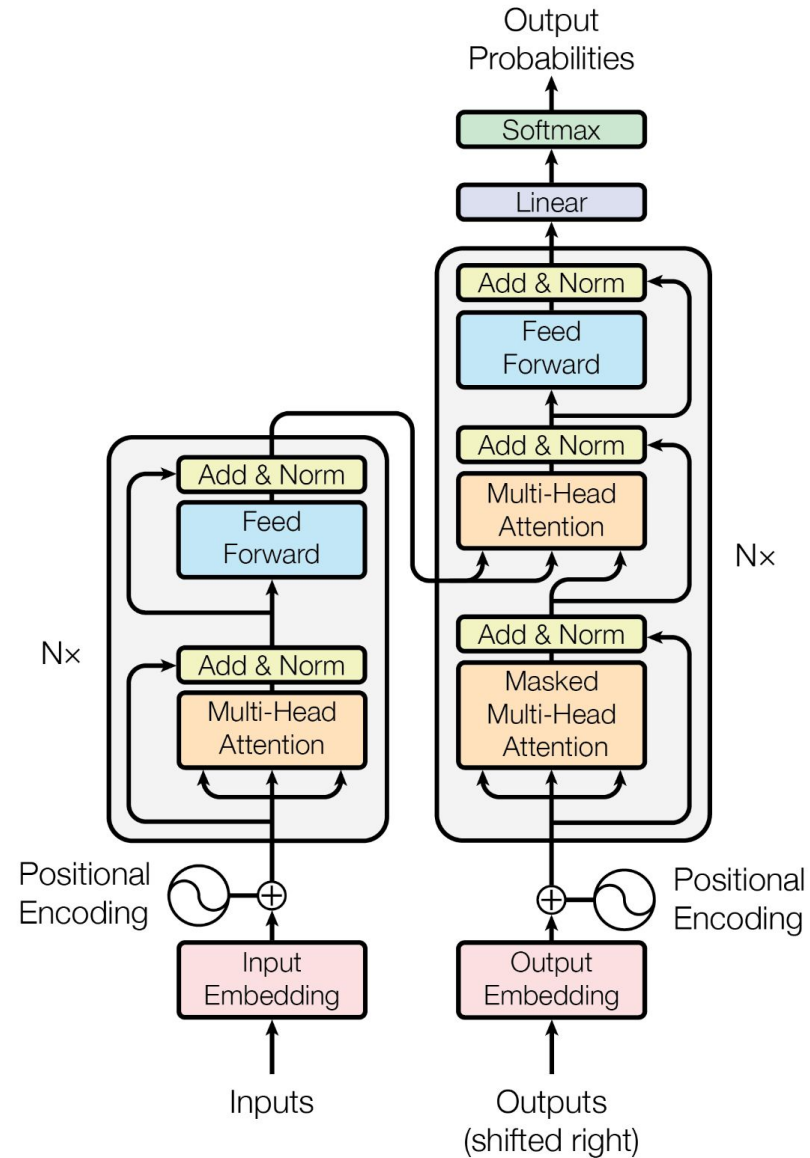
Scaled Dot-Product Attention



Multi-Head Attention



LLM evolution: Transformer (encoder - decoder)

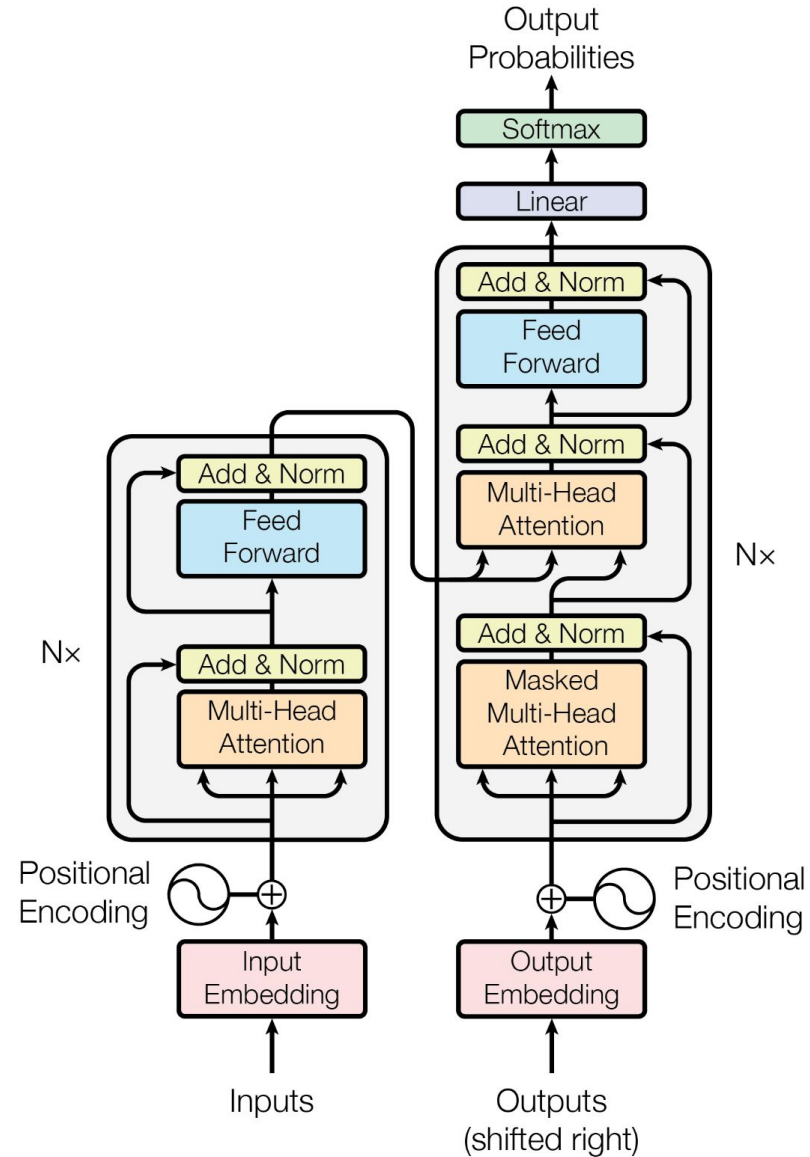


Vaswani *et al.*, 2017, Attention is all you need

LLM evolution: Transformer (encoder - decoder)



Massive Stacking



?



When does LLM stop generation?

Conclusions



Tokenization: context encoding

Transformer: parallelization, efficient learning

Fatty encoders: scaling (compute, data)



<https://en.uit.no/enhet/ifi>

Thank You

Arctic LLM Workshop 2023
Dept. of Computer Science



www.bioailab.org