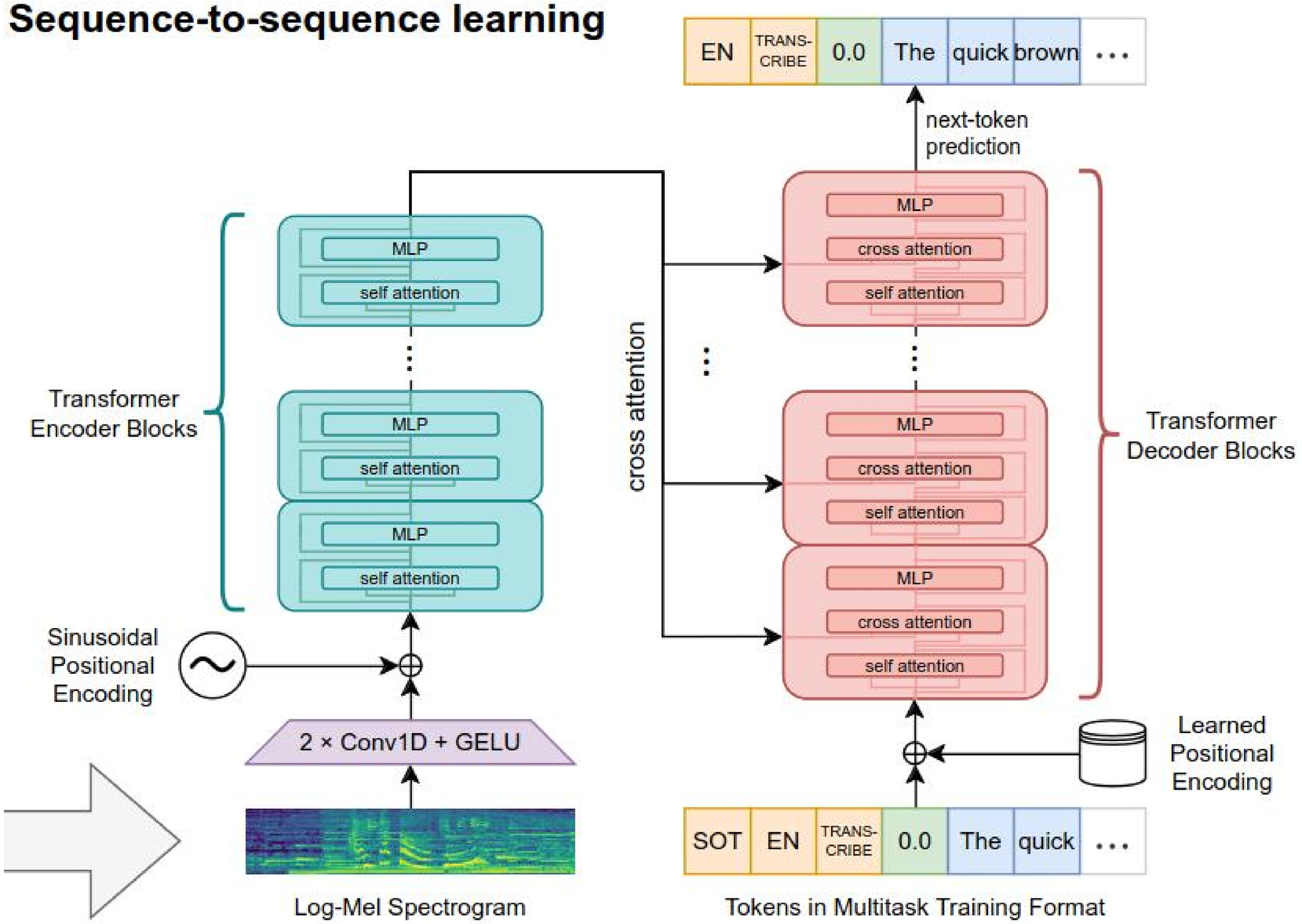
Understanding Whisper

Contents

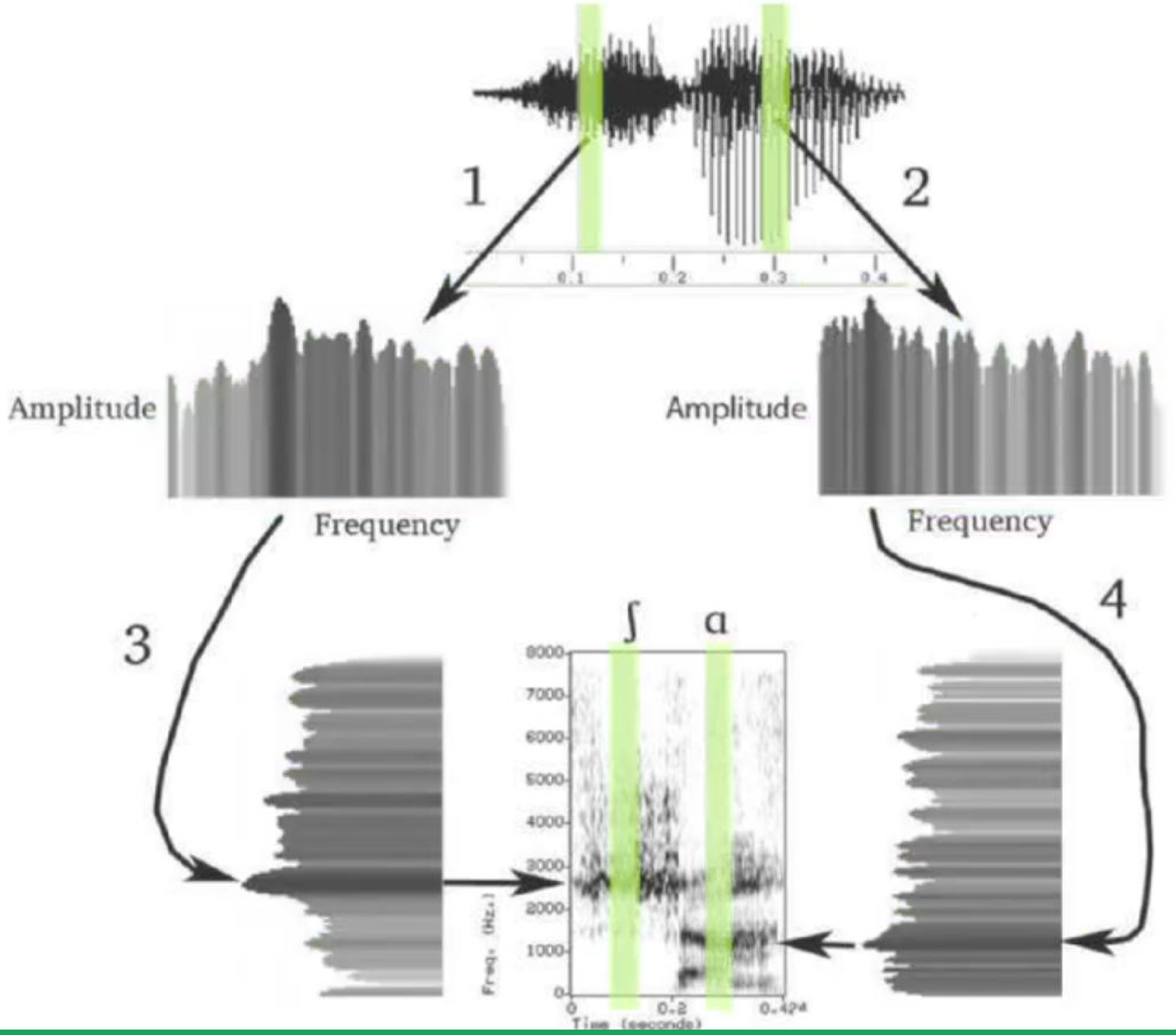
- Architecture
- Vocabulary and Output Format
- Demo

Architecture

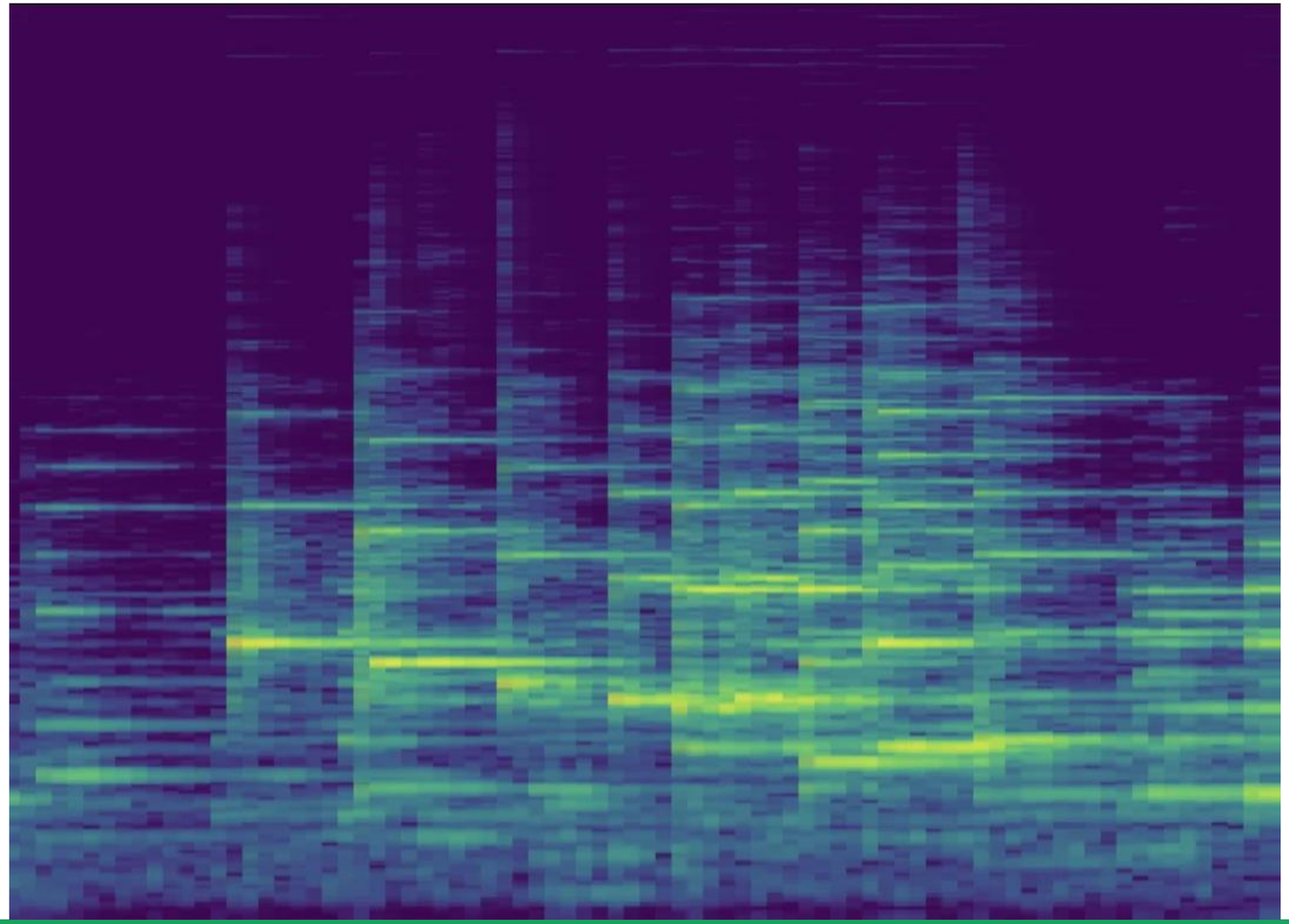
Overview


Sequence-to-sequence learning
EN
TRANS-CRIBE
0.0
The
quick
brown
...
next-token prediction
MLP
cross attention
self attention
cross attention
Transformer Encoder Blocks
Transformer Decoder Blocks
Sinusoidal Positional Encoding
2 × Conv1D + GELU
Learned Positional Encoding
SOT
EN
TRANS-CRIBE
0.0
The
quick
...
Log-Mel Spectrogram
Tokens in Multitask Training Format

Spectrogram



Spectrogram



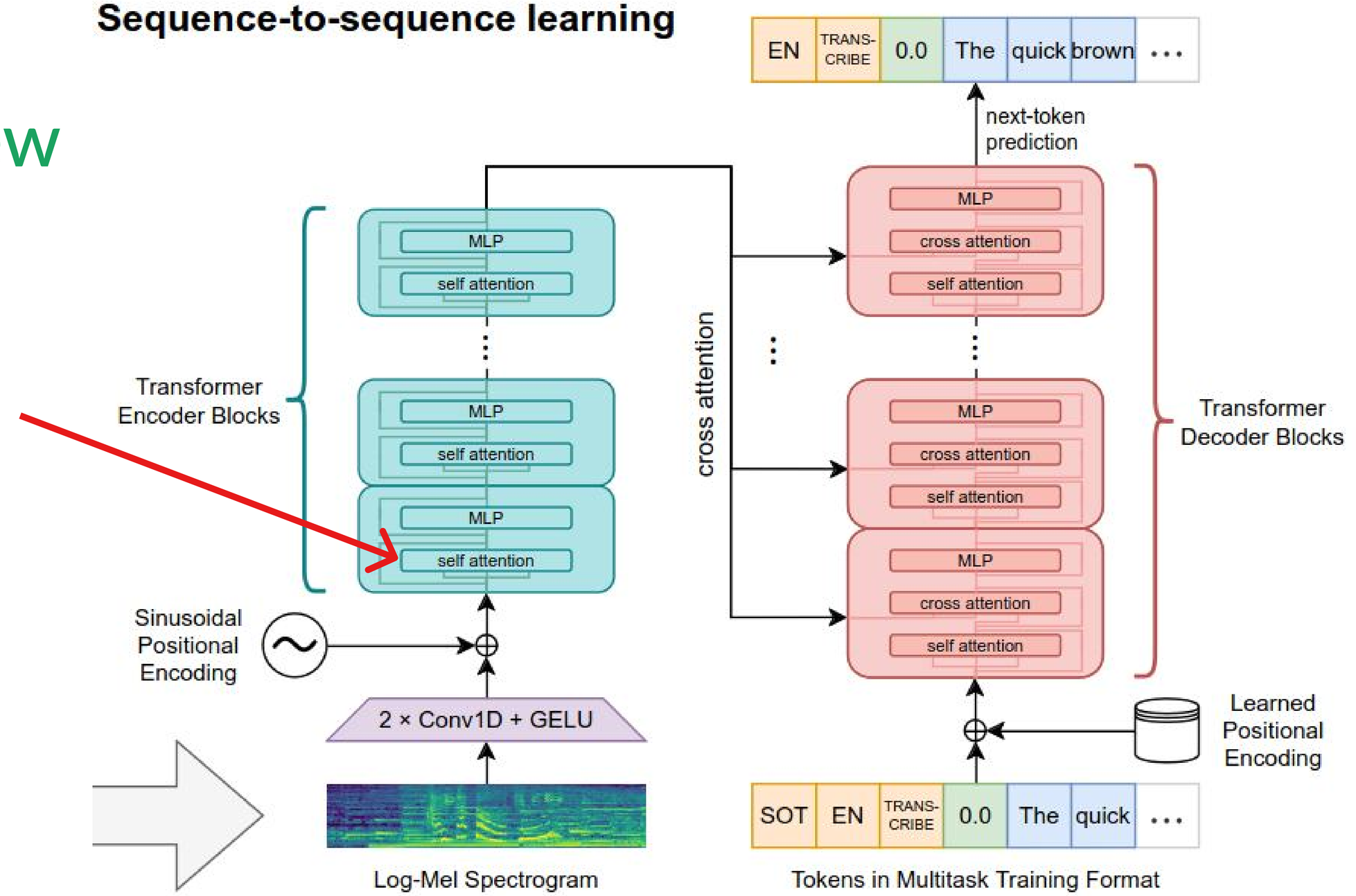
Whisper input

1.2	3.4	-100	5.3
3.9	10.7	3.2	12.3
-10.2	-29.5	9.8	14.2
.....	...	...	...
100.98	-22.4	-7.6	19.2

…

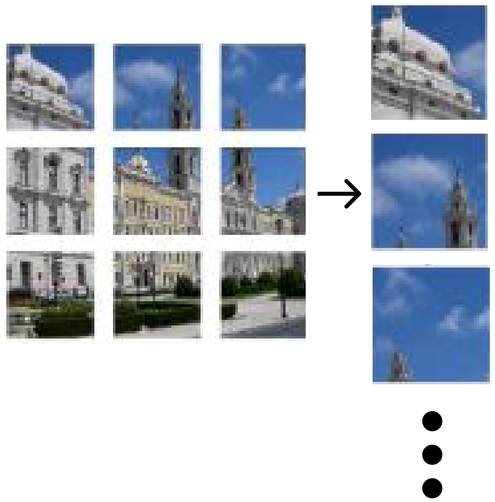
- The input is a set of vectors, one for each column of the spectrogram

Overview

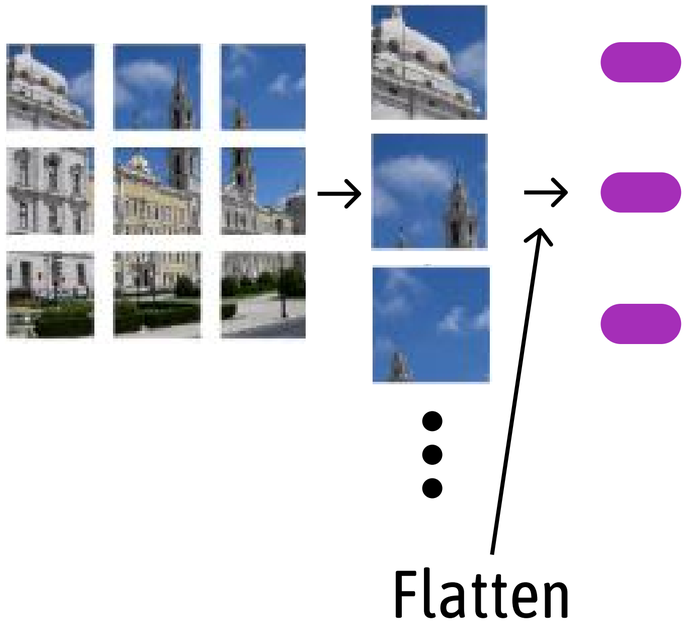

Sequence-to-sequence learning
EN
TRANS-CRIBE
0.0
The
quick
brown
...
next-token prediction
MLP
cross attention
self attention
Transformer Encoder Blocks
cross attention
Transformer Decoder Blocks
Sinusoidal Positional Encoding
2 × Conv1D + GELU
Learned Positional Encoding
SOT
EN
TRANS-CRIBE
0.0
The
quick
Log-Mel Spectrogram
Tokens in Multitask Training Format

Core Computational Block - Self attention

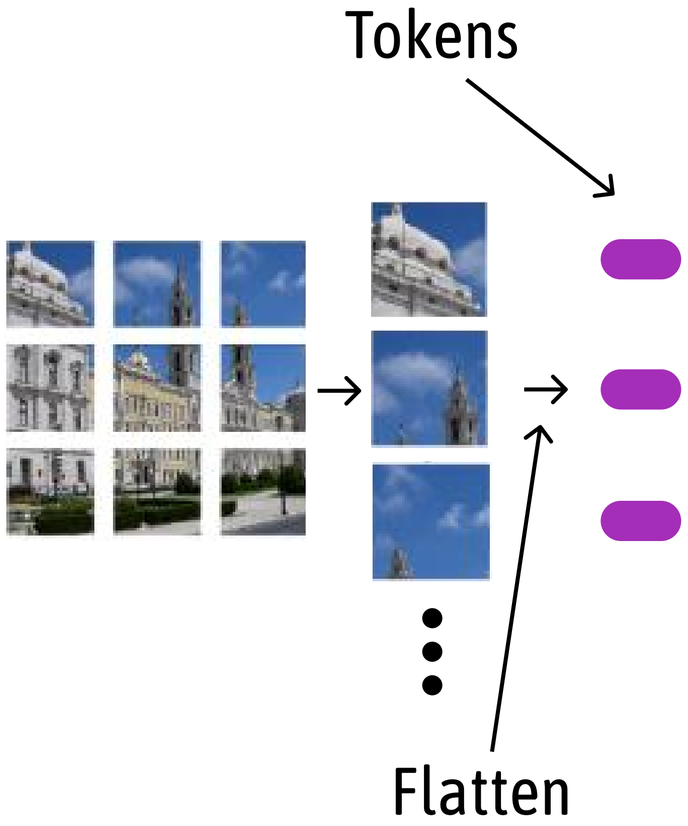
Core Computational Block - Self attention



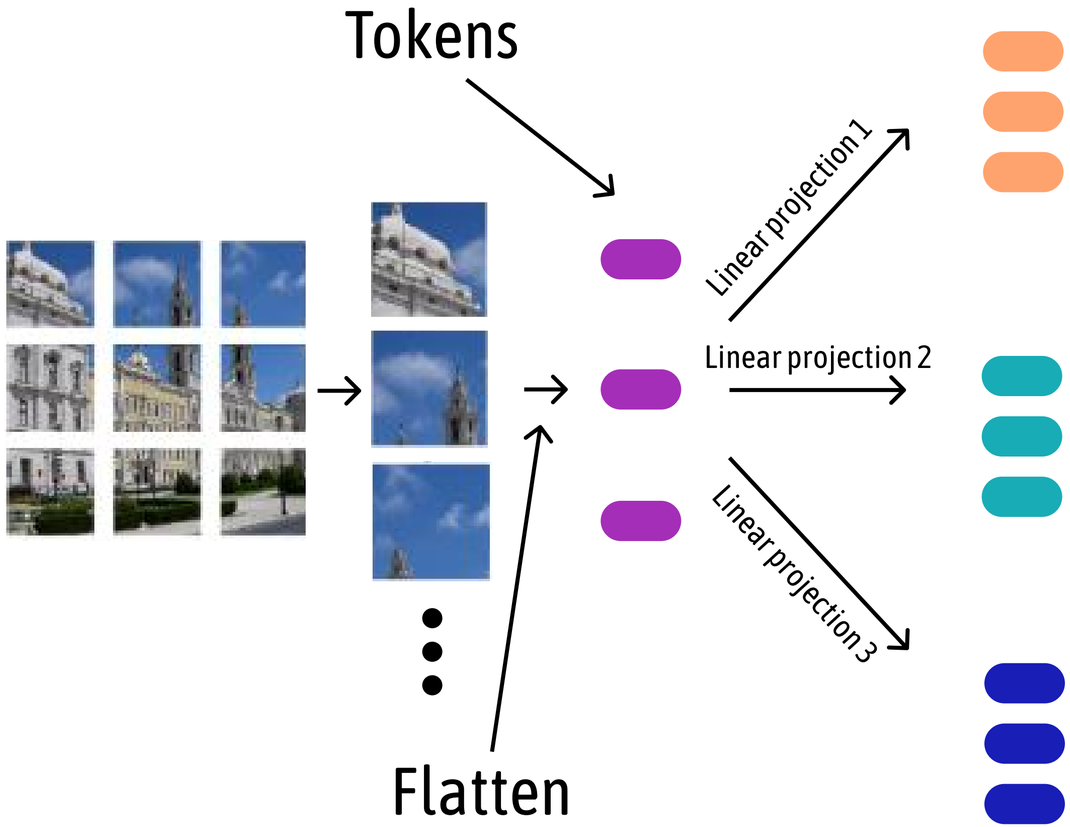
Core Computational Block - Self attention



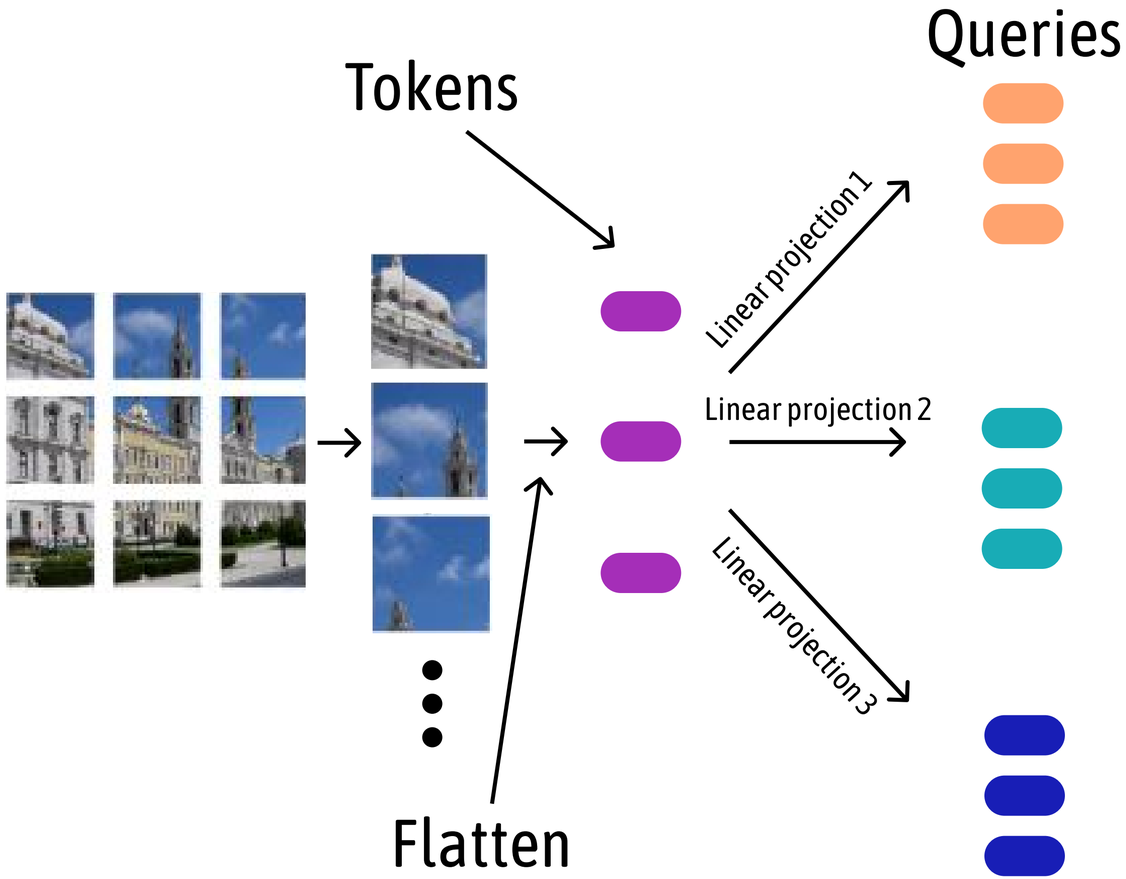
Core Computational Block - Self attention



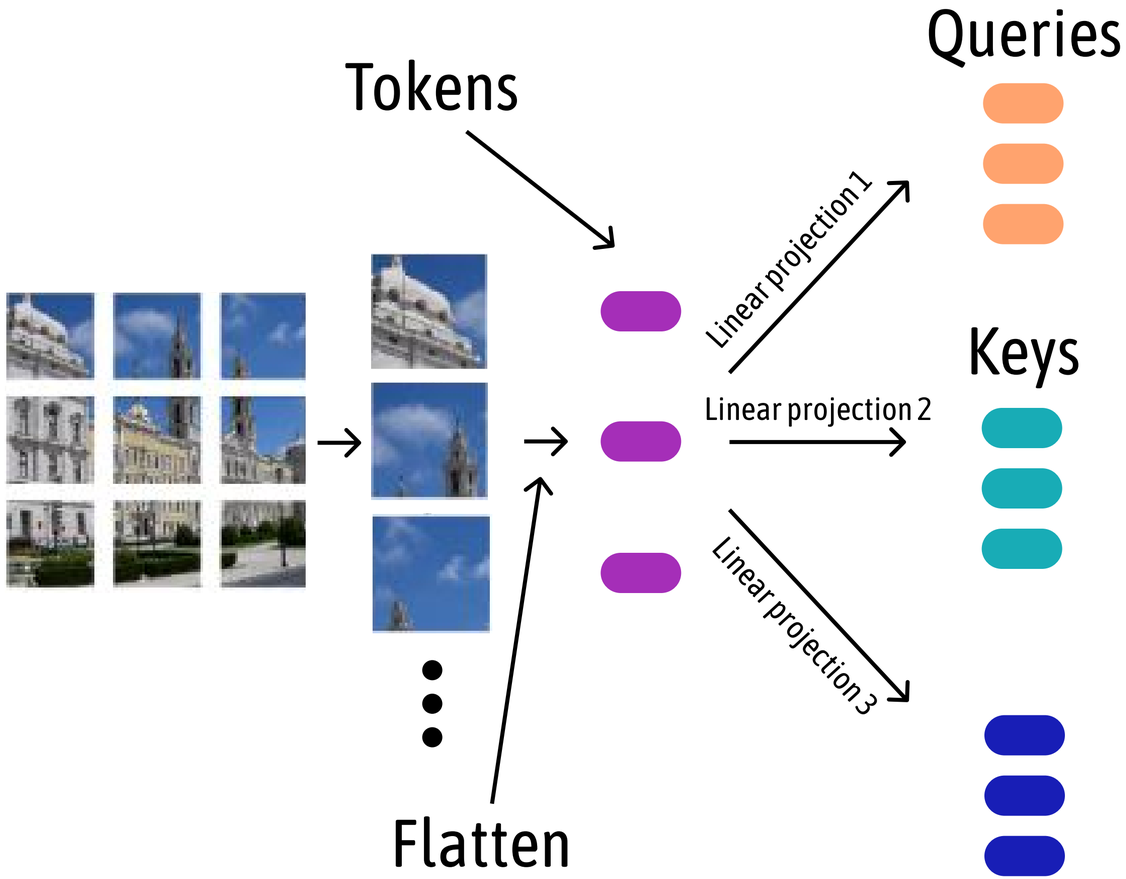
Core Computational Block - Self attention



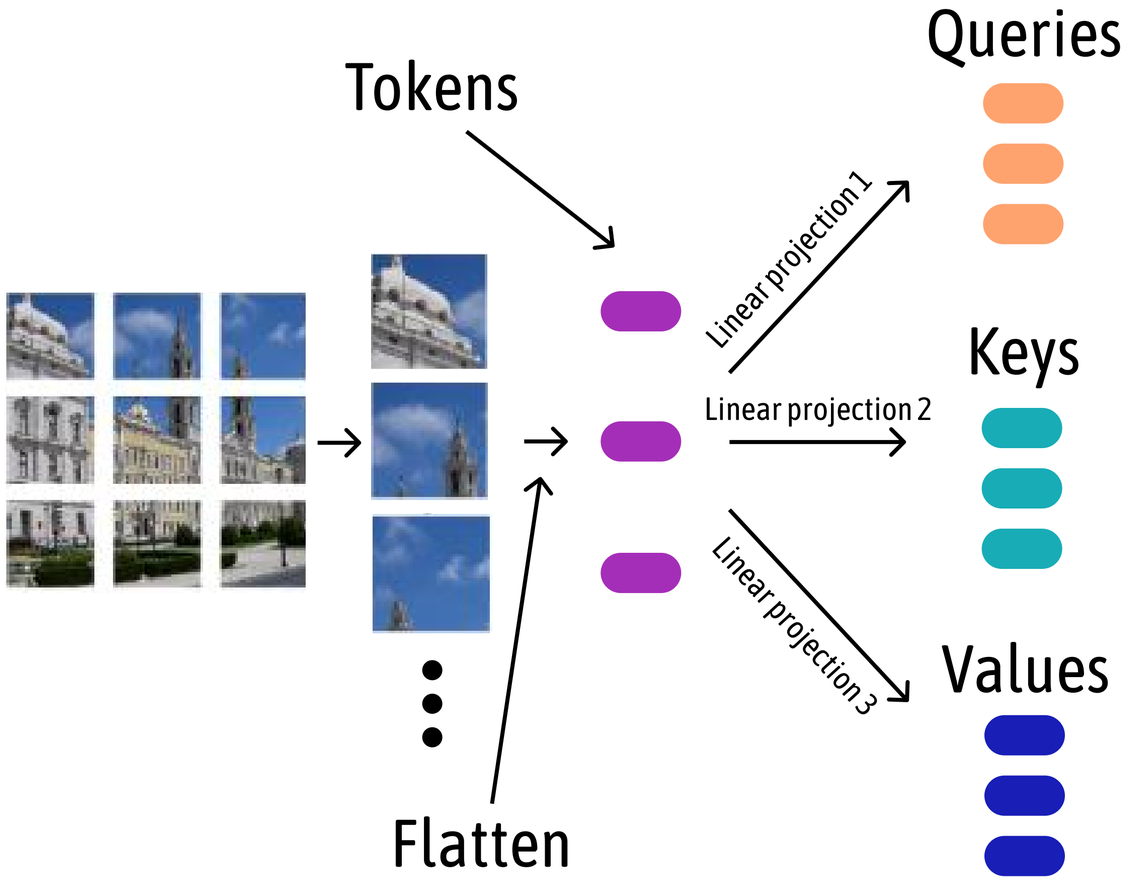
Core Computational Block - Self attention



Core Computational Block - Self attention

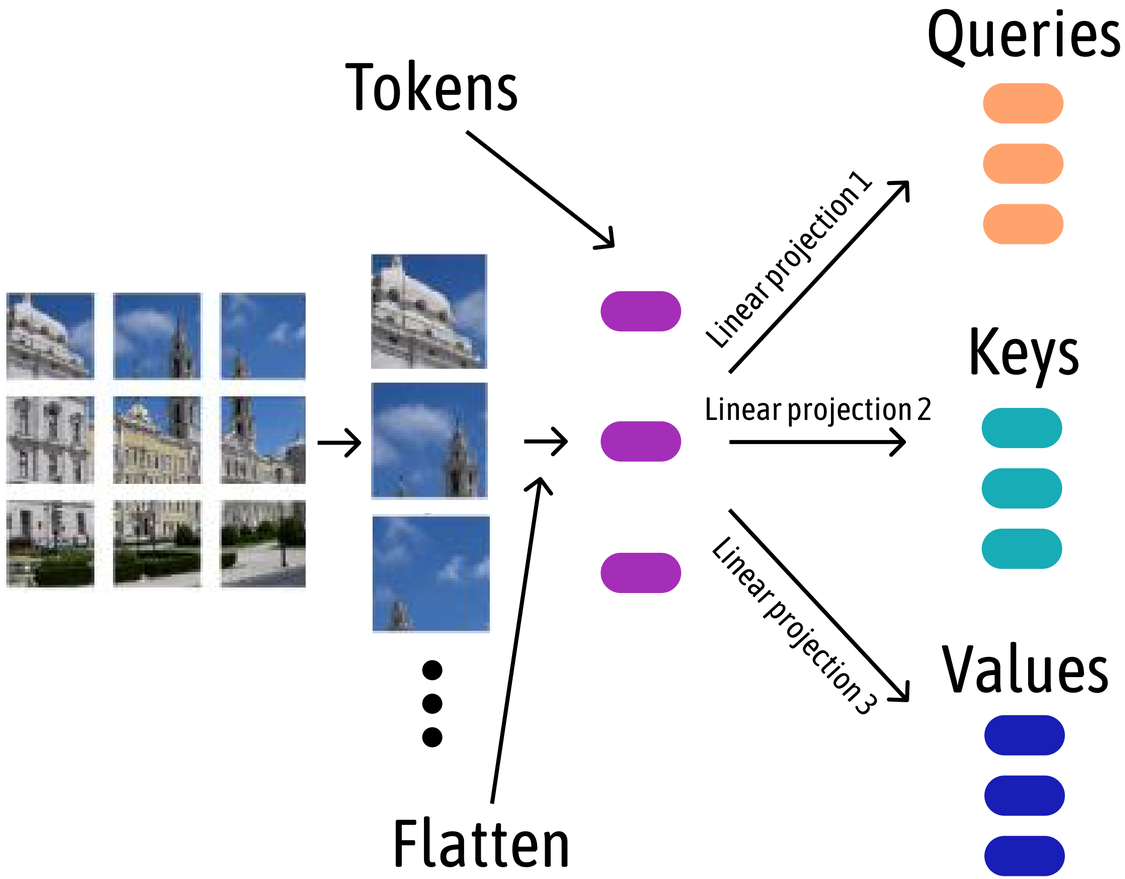


Core Computational Block - Self attention

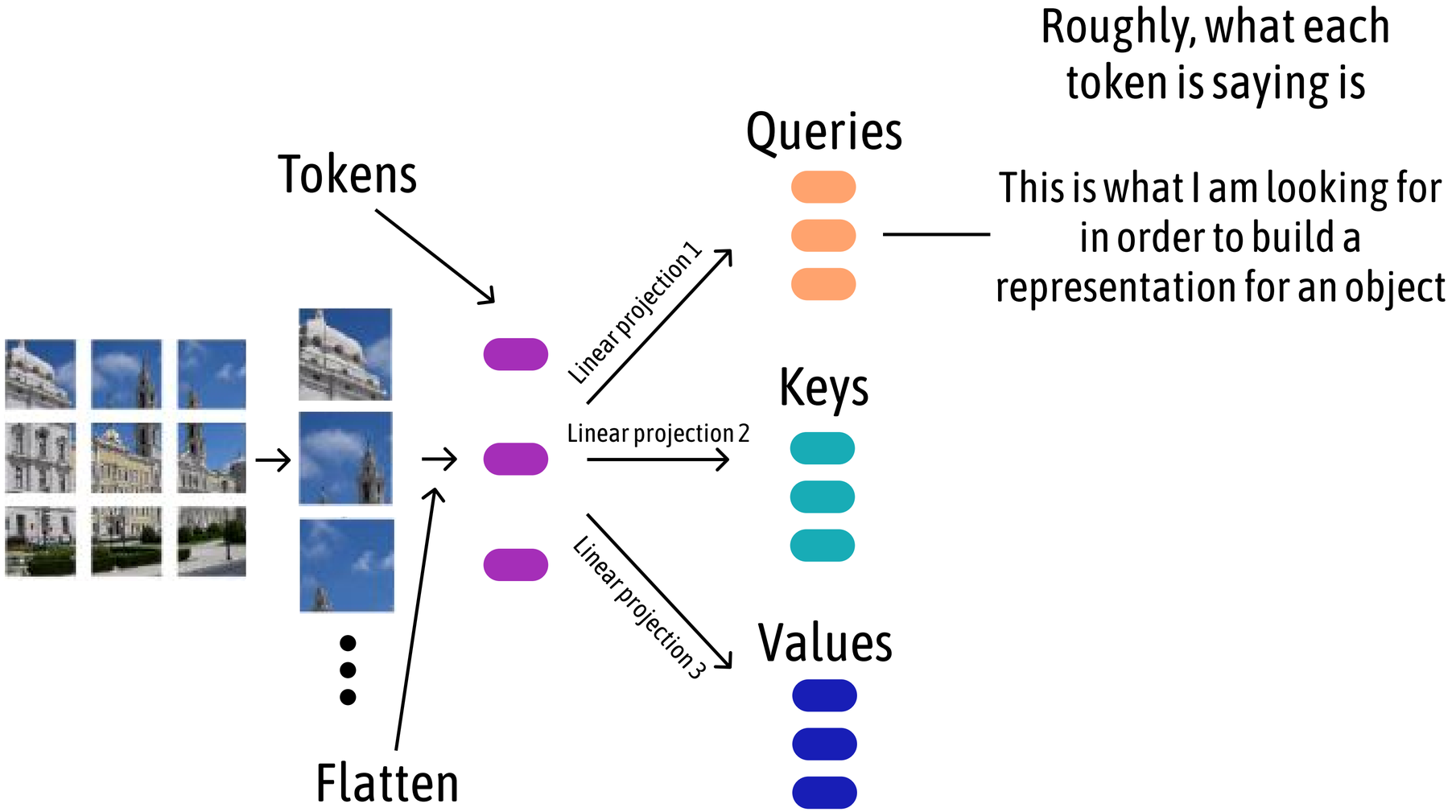


Core Computational Block - Self attention

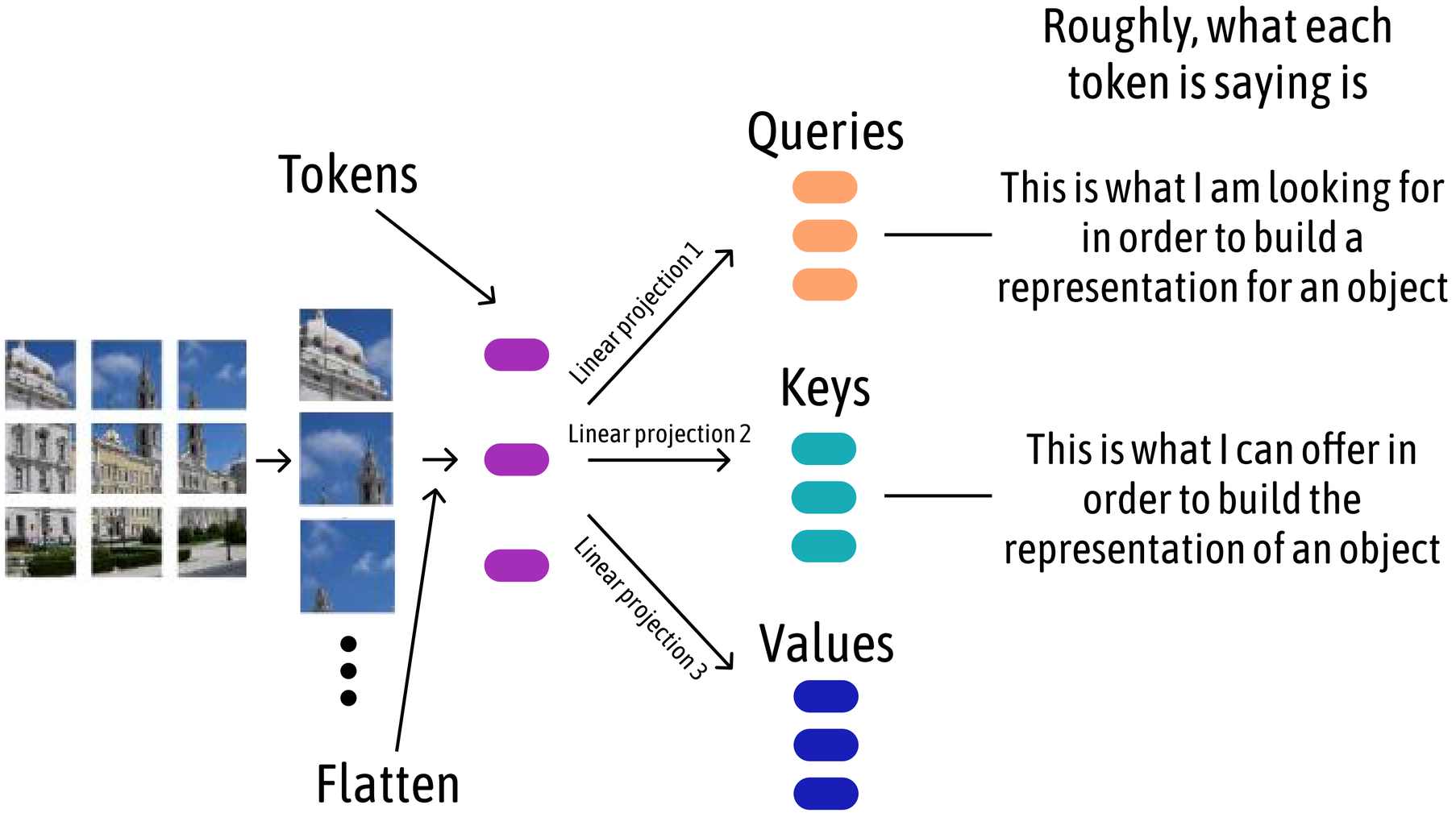
Roughly, what each token is saying is



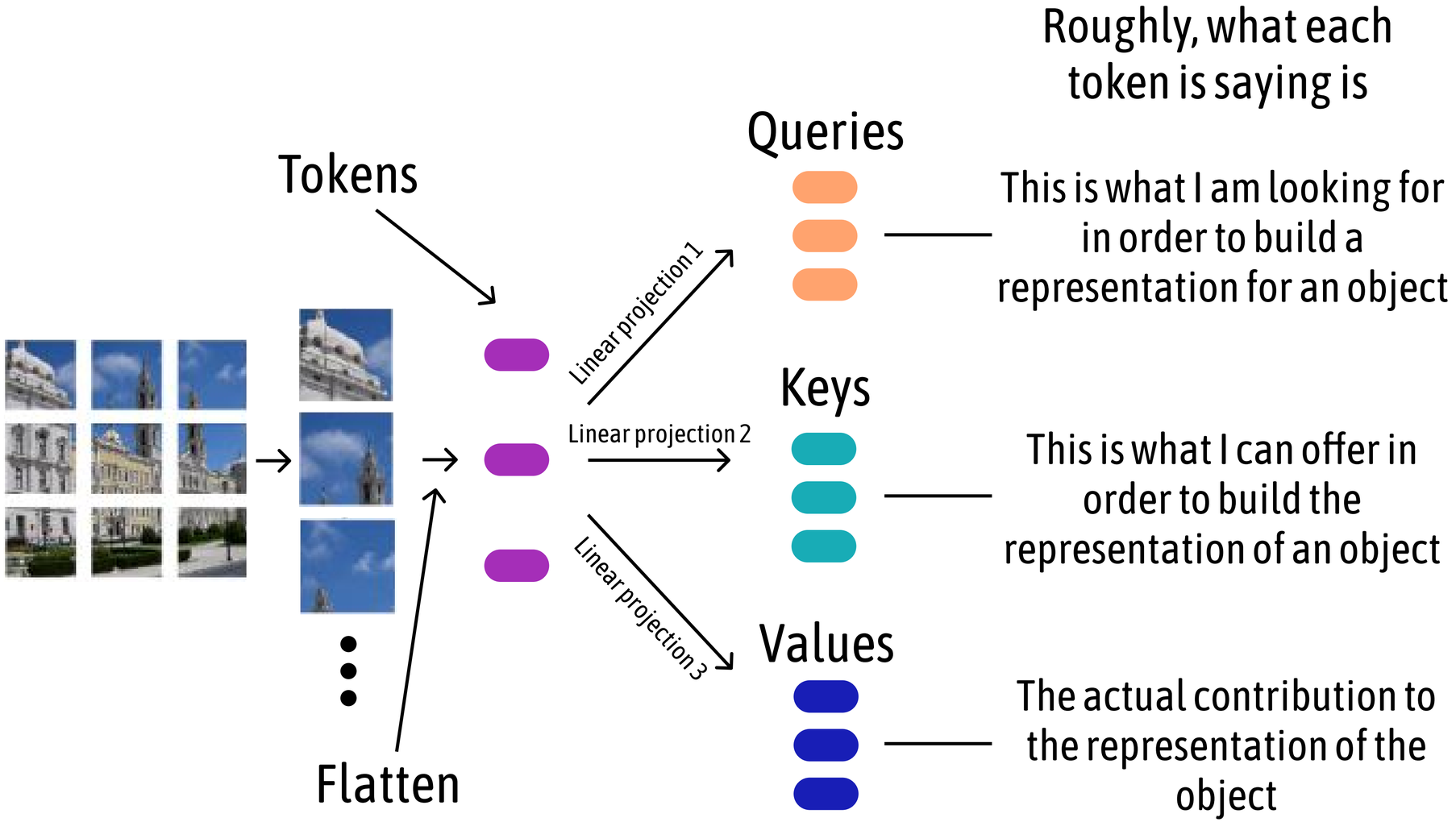
Core Computational Block - Self attention



Core Computational Block - Self attention



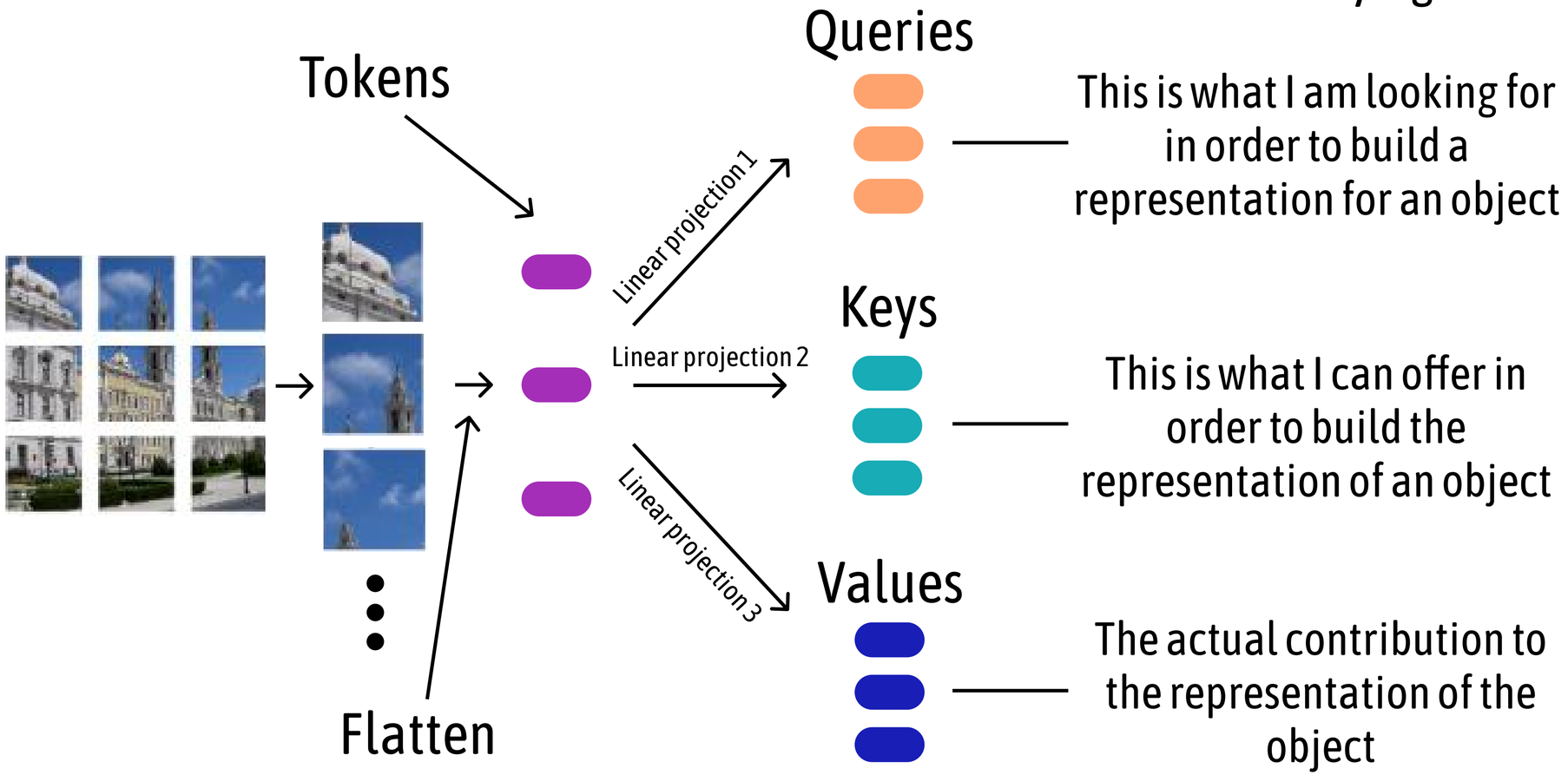
Core Computational Block - Self attention



Core Computational Block - Self attention

Roughly, what each token is saying is

Example



Core Computational Block - Self attention

Flatten

Tokens

Linear projection 1

Linear projection 2

Linear projection 3

Roughly, what each token is saying is

Example

Queries

This is what I am looking for in order to build a representation for an object

I am looking to build the representation of a tall brown chair

Keys

This is what I can offer in order to build the representation of an object

Values

The actual contribution to the representation of the object

Core Computational Block - Self attention

Flatten

Tokens

Linear projection 1

Linear projection 2

Linear projection 3

Queries

Keys

Values

Roughly, what each token is saying is

This is what I am looking for in order to build a representation for an object

This is what I can offer in order to build the representation of an object

The actual contribution to the representation of the object

Example

I am looking to build the representation of a tall brown chair

I can offer the representation of a leg for a tall brown chair

Core Computational Block - Self attention

Flatten

Tokens

Linear projection 1

Linear projection 2

Linear projection 3

Queries

Keys

Values

Roughly, what each token is saying is

Example

This is what I am looking for in order to build a representation for an object

I am looking to build the representation of a tall brown chair

This is what I can offer in order to build the representation of an object

I can offer the representation of a leg for a tall brown chair

The actual contribution to the representation of the object

The actual representation of the chair leg

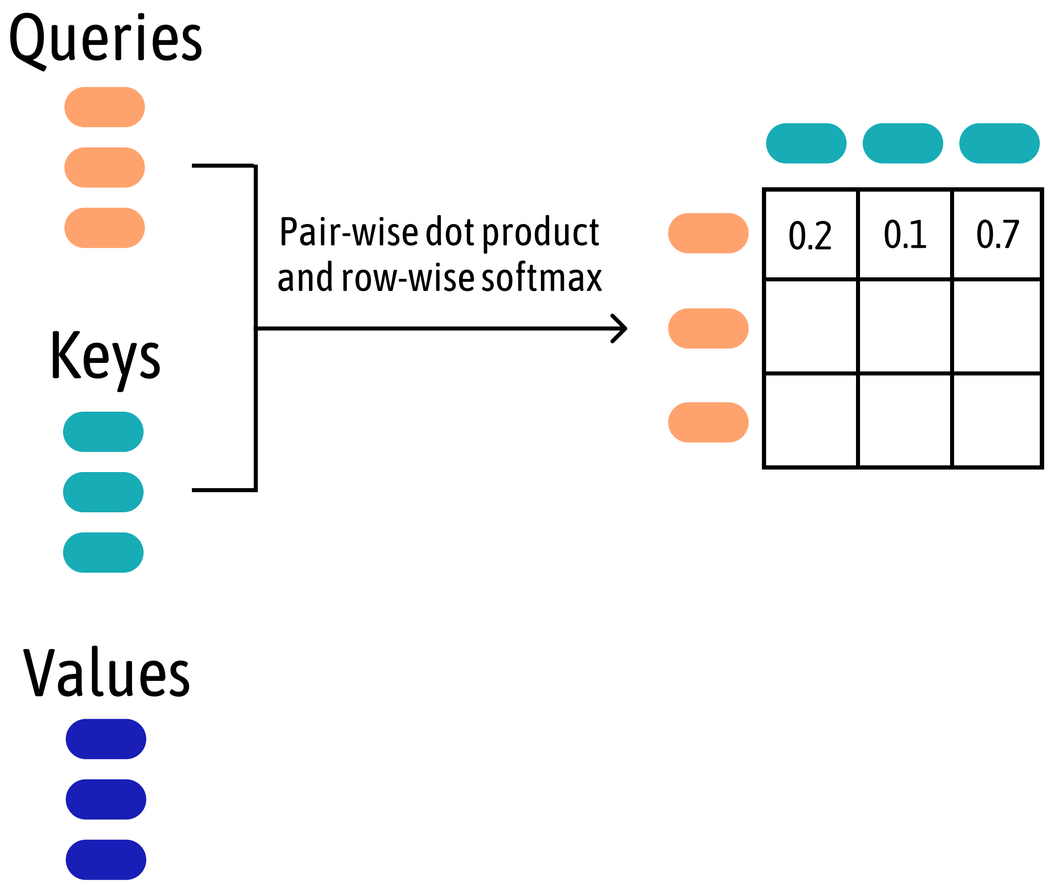
Core Computational Block - Self attention

Queries

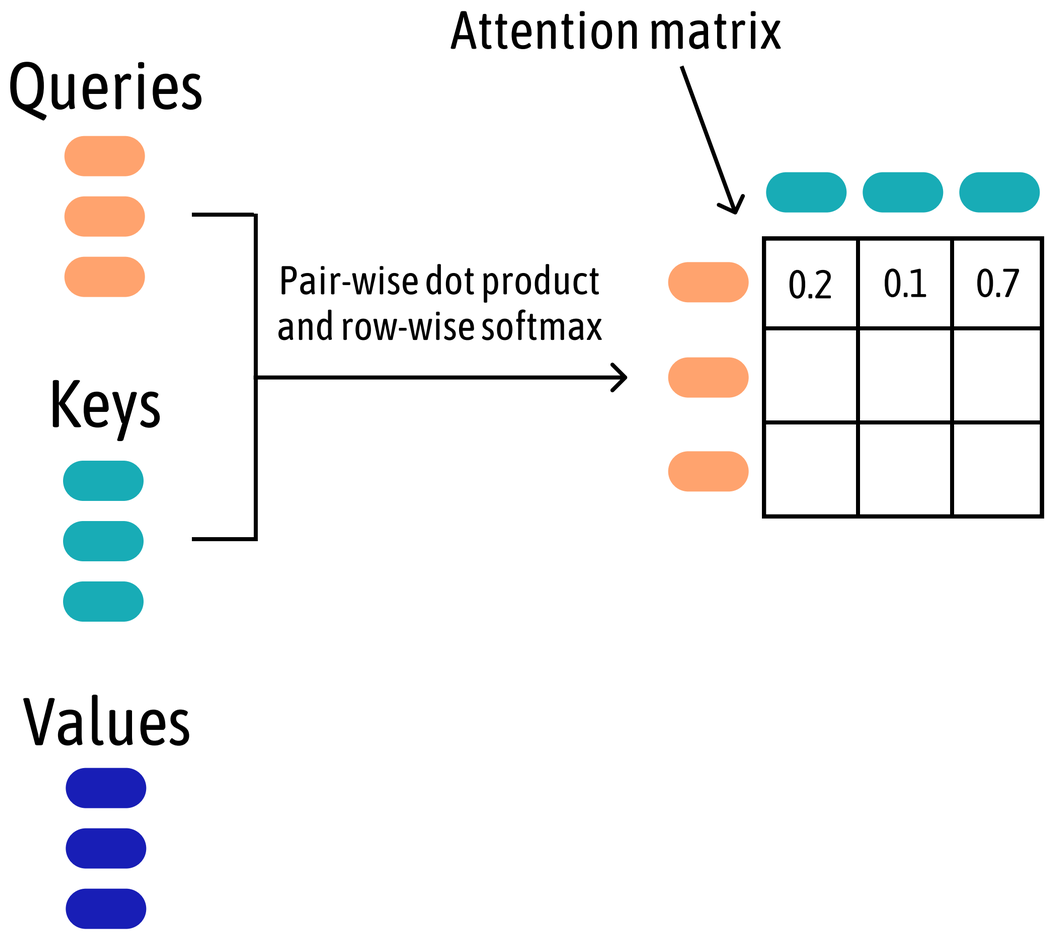
Keys

Values

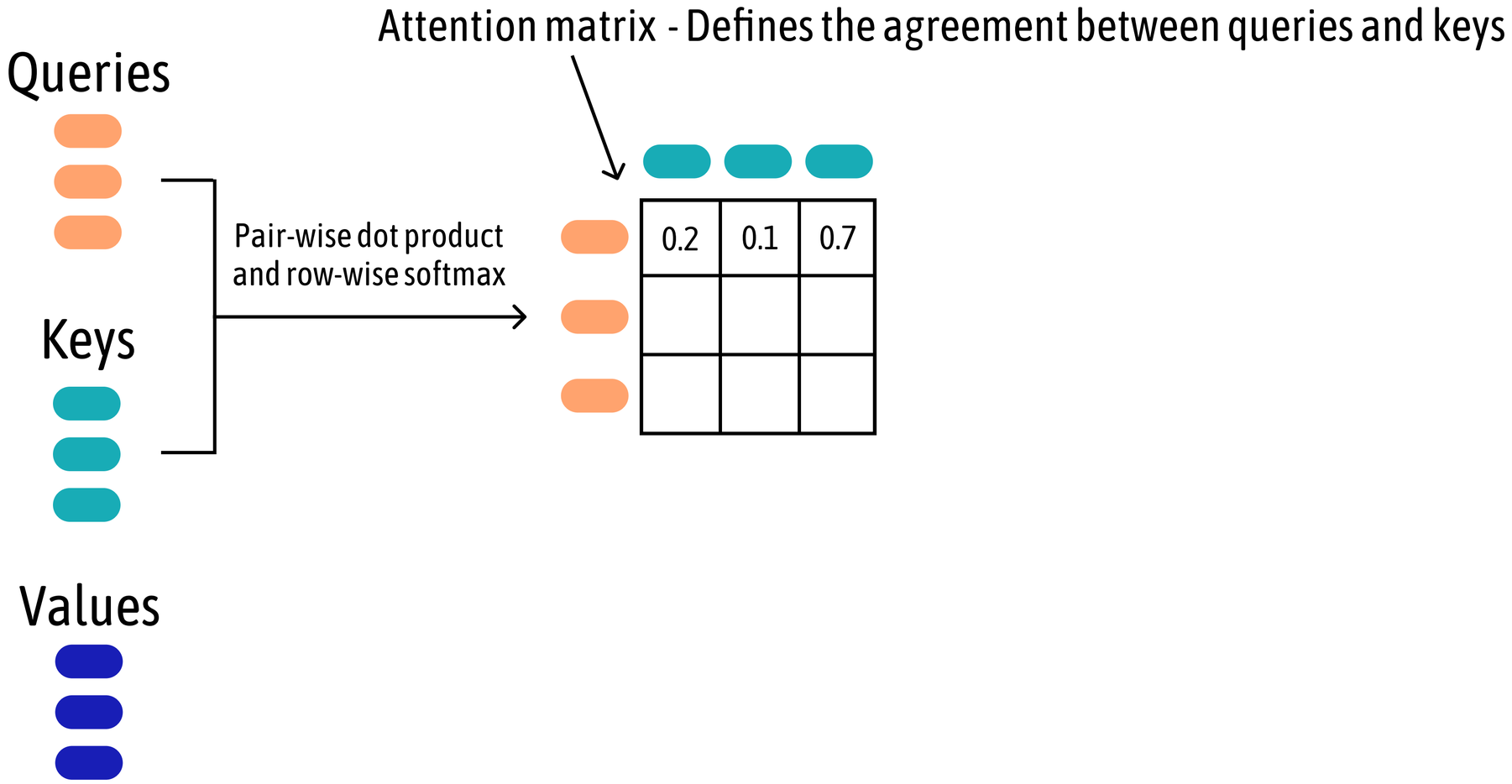
Core Computational Block - Self attention



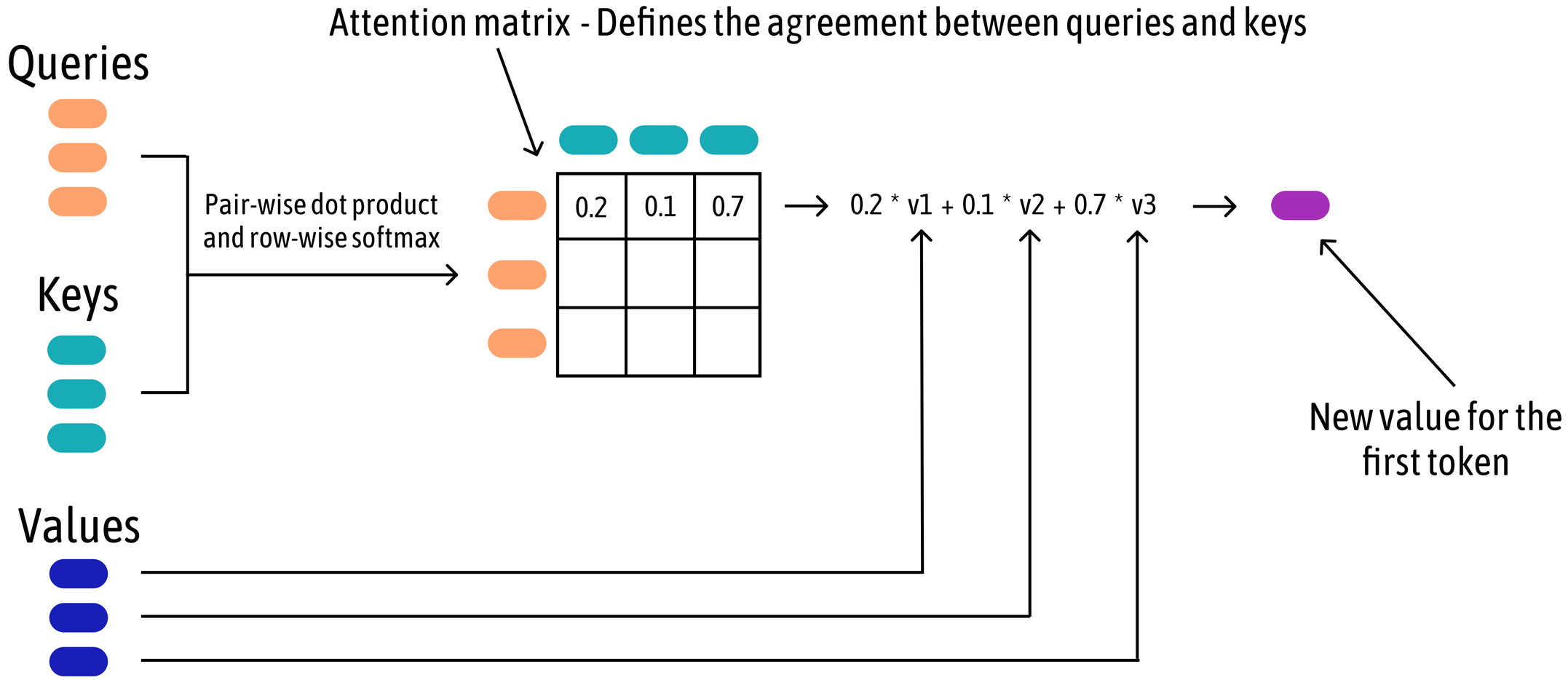
Core Computational Block - Self attention



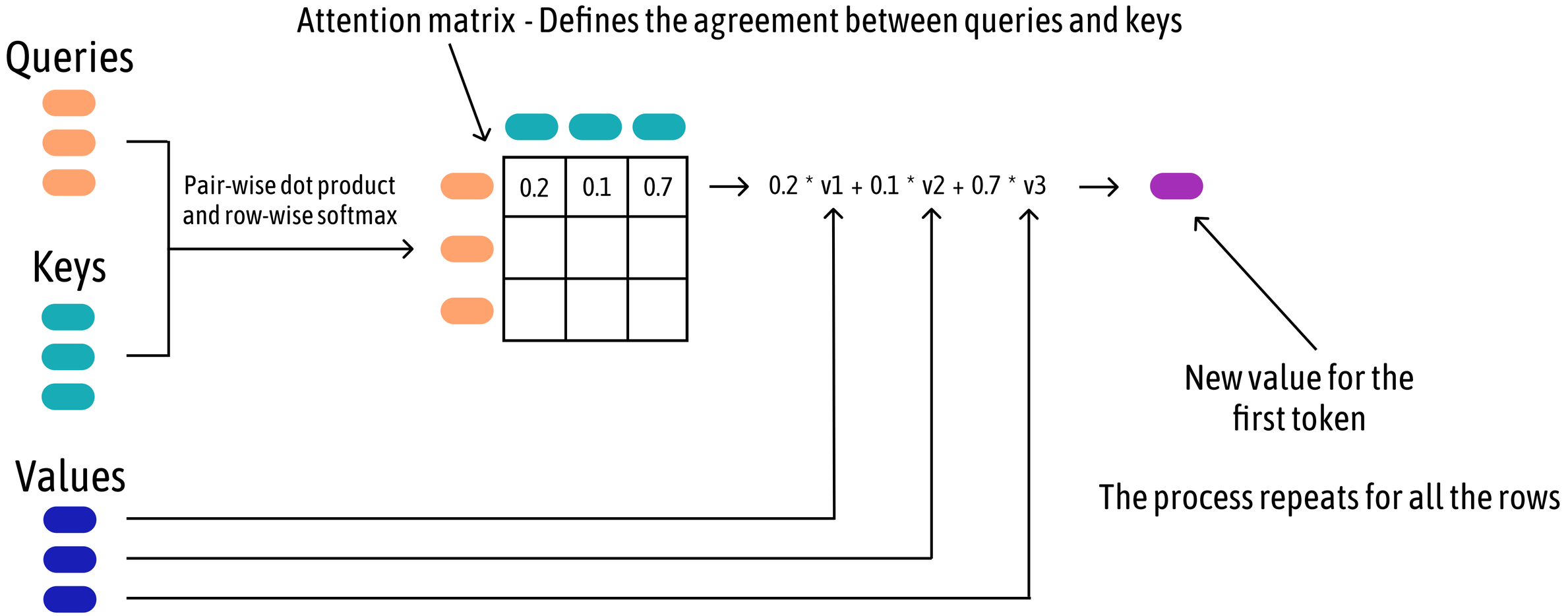
Core Computational Block - Self attention



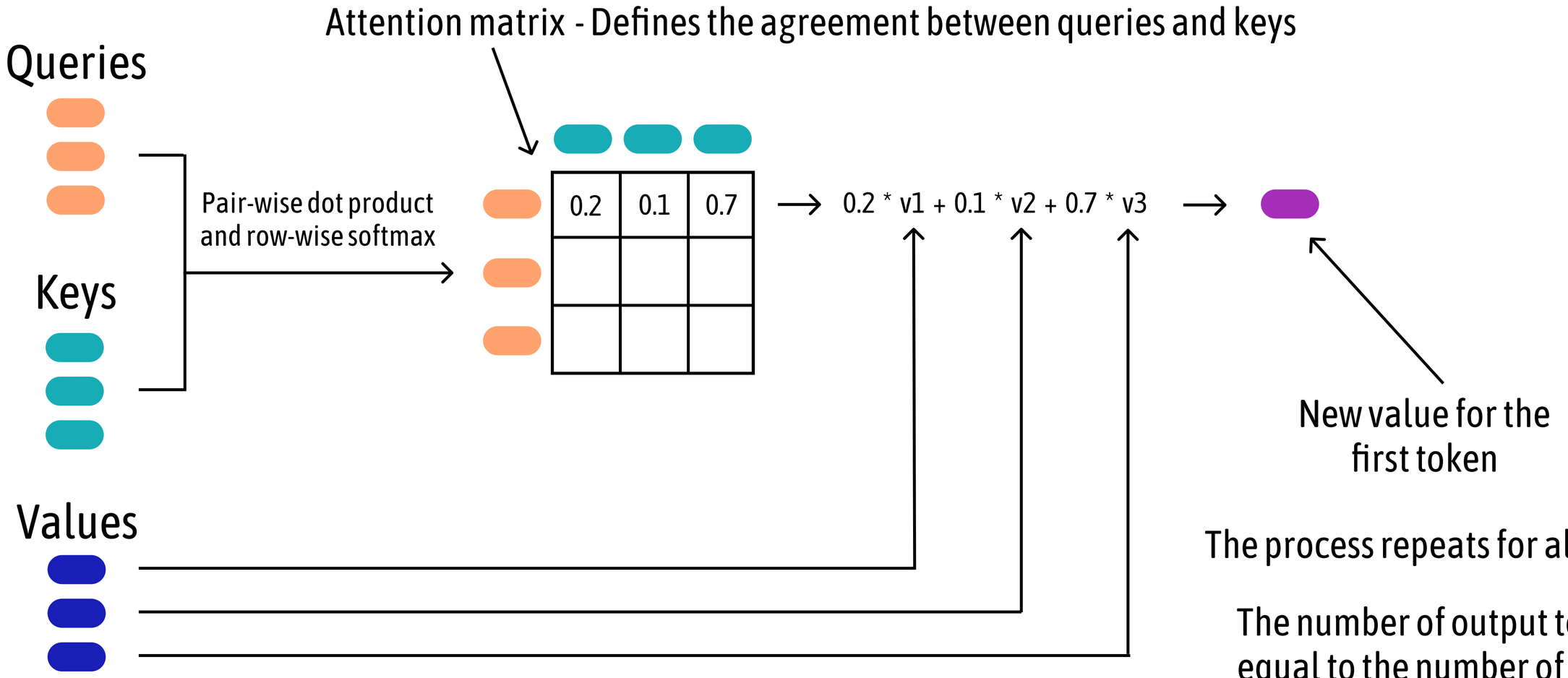
Core Computational Block - Self attention



Core Computational Block - Self attention



Core Computational Block - Self attention



The process repeats for all the rows

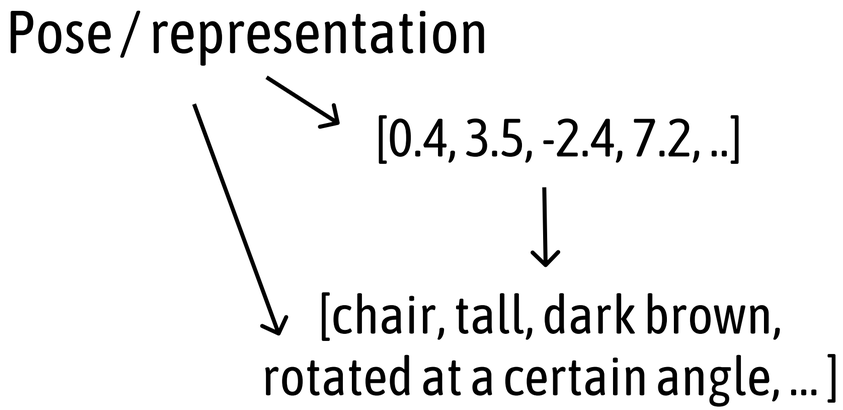
The number of output tokens is equal to the number of queries

Computer graphics - Rendering

Pose / representation

[0.4, 3.5, -2.4, 7.2, ..]

Computer graphics - Rendering



Computer graphics - Rendering

Pose / representation

[0.4, 3.5, -2.4, 7.2, ..]

[chair, tall, dark brown,
rotated at a certain angle, ...]

Computer graphics - Rendering

Pose / representation

[0.4, 3.5, -2.4, 7.2, ..] Linear Transformation

[chair, tall, dark brown,
rotated at a certain angle, ...]

Computer graphics - Rendering

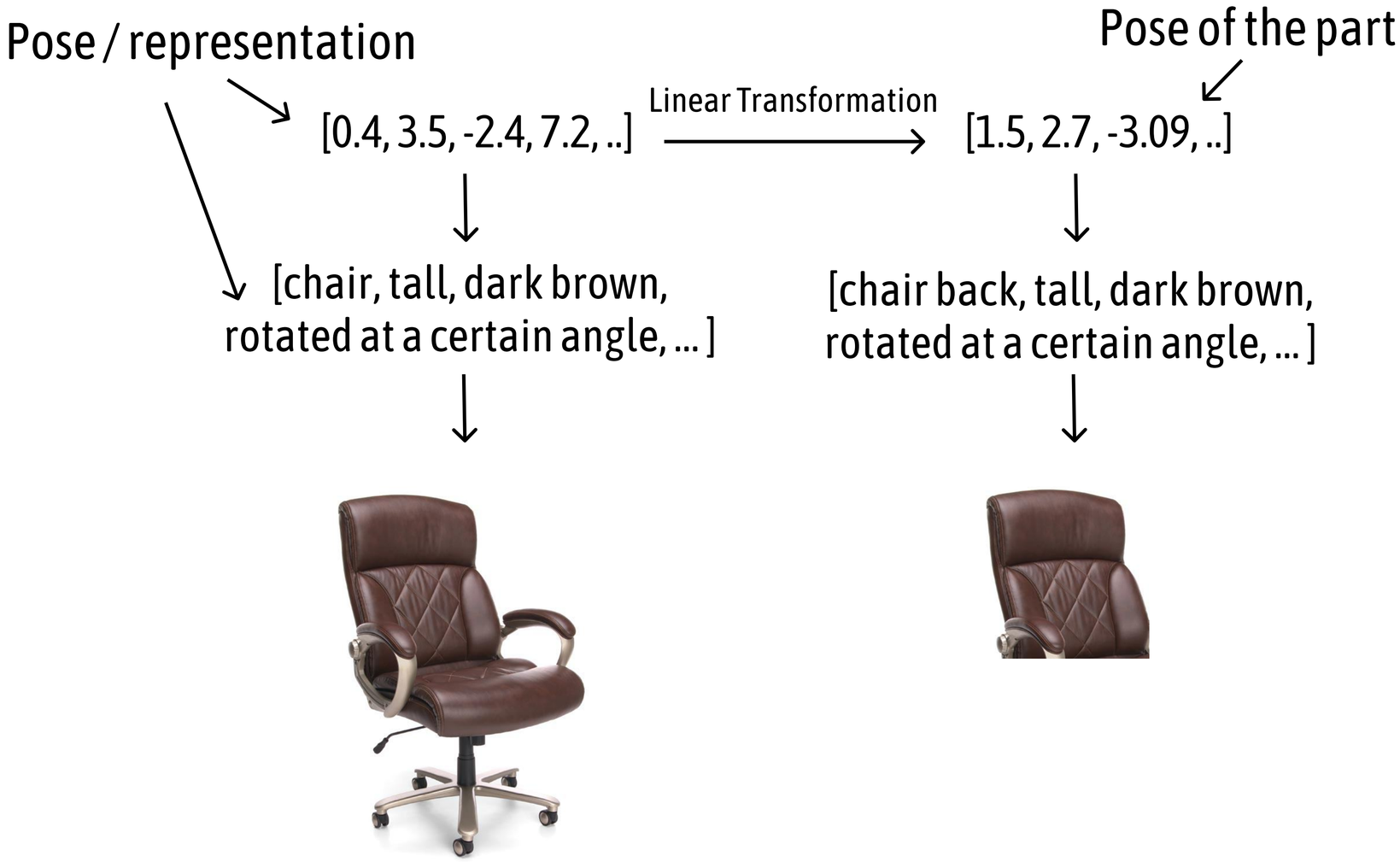
Pose / representation

Pose of the part

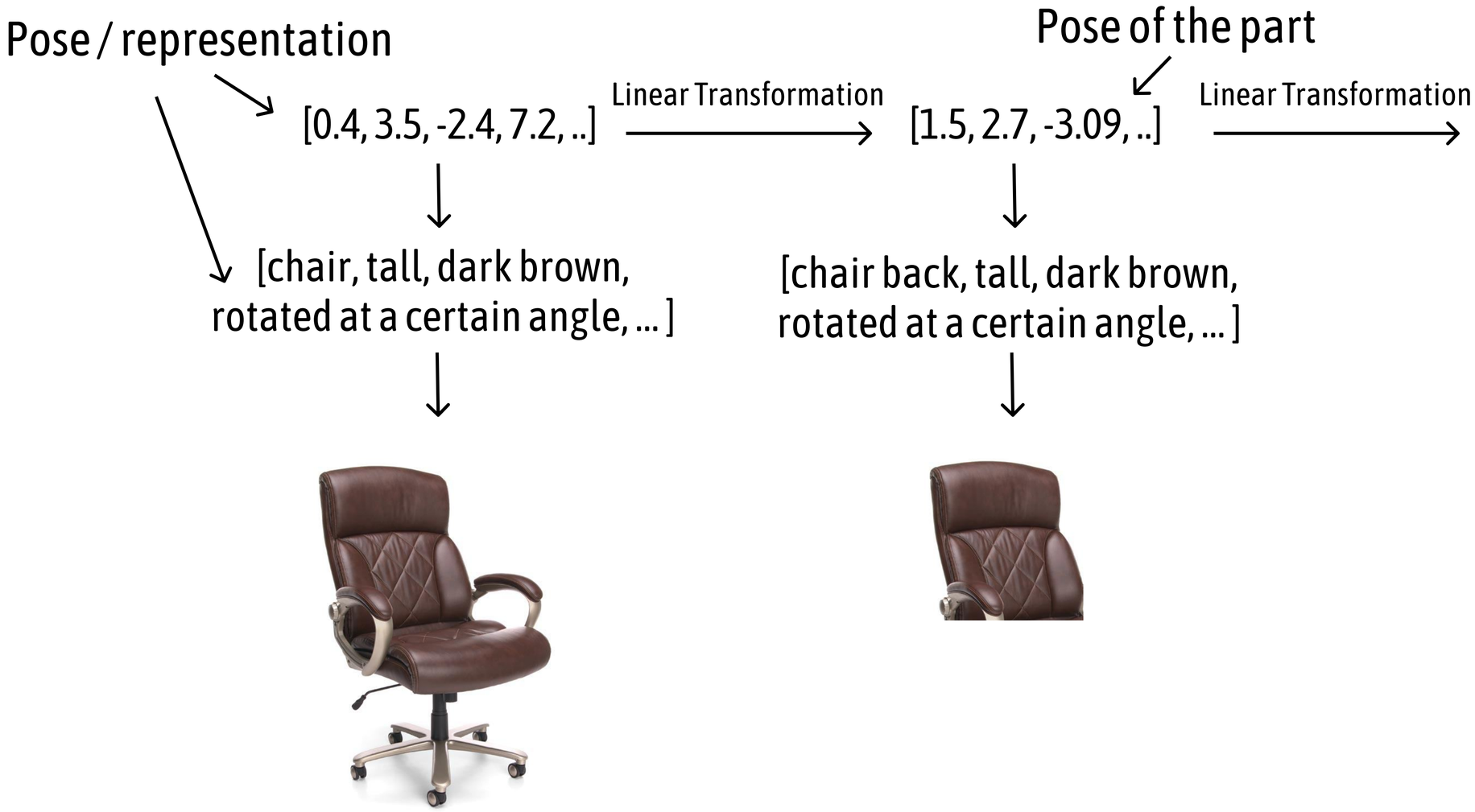
[0.4, 3.5, -2.4, 7.2, ..] Linear Transformation [1.5, 2.7, -3.09, ..]

[chair, tall, dark brown,
rotated at a certain angle, ...]

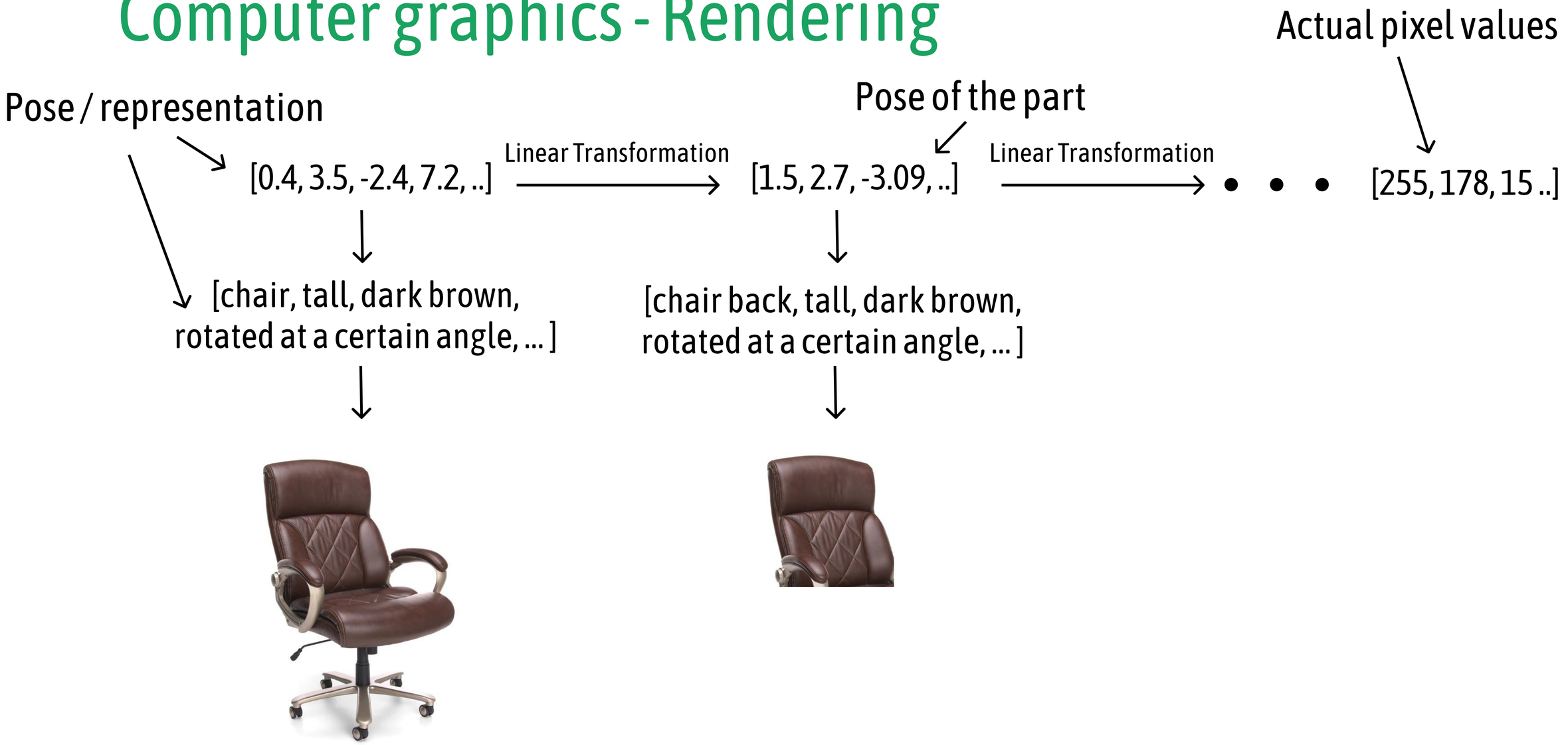
Computer graphics - Rendering



Computer graphics - Rendering



Computer graphics - Rendering



Computer graphics - Rendering

Pose / representation

[0.4, 3.5, -2.4, 7.2, ..]

Linear Transformation

Pose of the part

[1.5, 2.7, -3.09, ..]

Linear Transformation

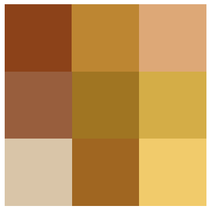
• • •

Actual pixel values

[255, 178, 15 ..]

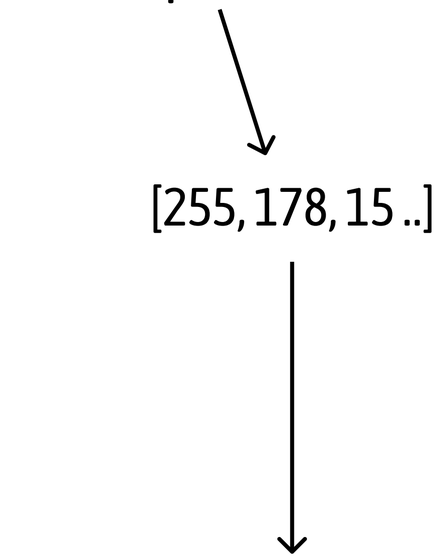
[chair, tall, dark brown, rotated at a certain angle, ...]

[chair back, tall, dark brown, rotated at a certain angle, ...]

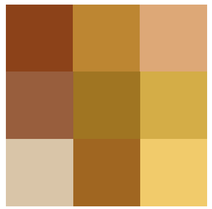


Computer vision - Derendering

Actual pixel values



[255, 178, 15 ..]

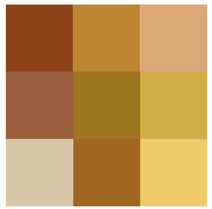


Computer vision - Derendering

Actual pixel values

[255, 178, 15 ..]

Linear Transformation



Try to learn the inverse linear transformation
Corresponding to the one that would have been used to render the object

Computer vision - Derendering

Actual pixel values

Estimate pose of the part

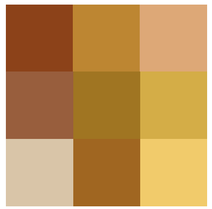
[255, 178, 15 ..]

Linear Transformation

• • •

[1.5, 2.7, -3.09, ..]

[chair back, tall, dark brown,
rotated at a certain angle, ...]



Computer vision - Derendering

Actual pixel values

[255, 178, 15 ..]

Linear Transformation

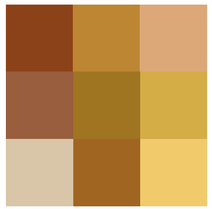
• • •

Estimate pose of the part

[1.5, 2.7, -3.09, ..]

Linear Transformation

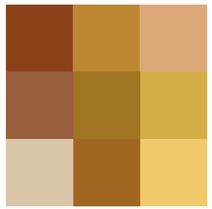
[chair back, tall, dark brown,
rotated at a certain angle, ...]



Computer vision - Derendering

Actual pixel values

[255, 178, 15 ..]



Linear Transformation

• • •

Estimate pose of the part

[1.5, 2.7, -3.09, ..]

[chair back, tall, dark brown, rotated at a certain angle, ...]

Linear Transformation

Estimate pose of the object

[0.4, 3.5, -2.4, 7.2, ..]

[chair, tall, dark brown, rotated at a certain angle, ...]

Implementing Derendering - Capsule networks

Implementing Derendering - Capsule networks

Start from the pose of the parts,
and try to build the pose of the whole

Implementing Derendering - Capsule networks

Linear Transformation

Linear Transformation

Linear Transformation

Linear Transformation

Implementing Derendering - Capsule networks

Linear Transformation

Linear Transformation

Linear Transformation

Linear Transformation

Implementing Derendering - Capsule networks

Linear Transformation

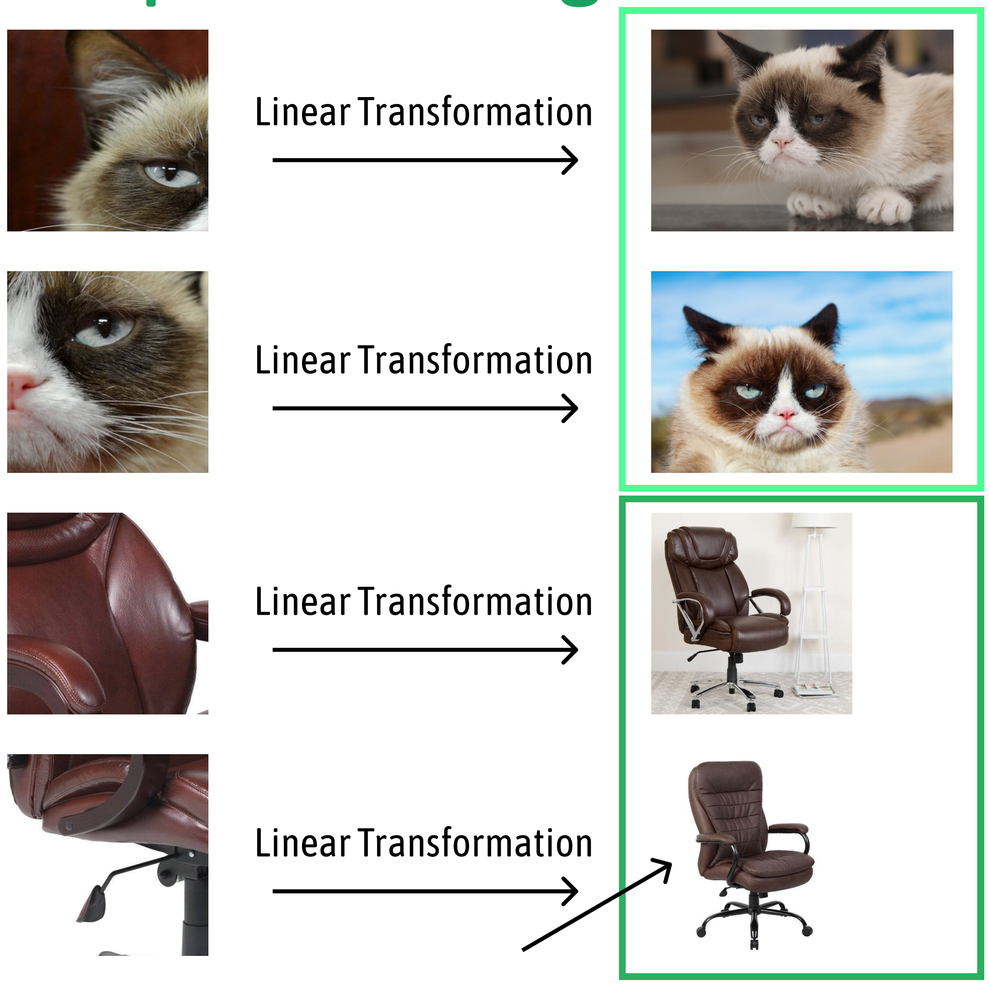
Linear Transformation

Linear Transformation

Linear Transformation

Estimated pose of the whole

Implementing Derendering - Capsule networks



Implementing Derendering - Capsule networks

Linear Transformation

Linear Transformation

Weighted avg.

Linear Transformation

Linear Transformation

Weighted avg.

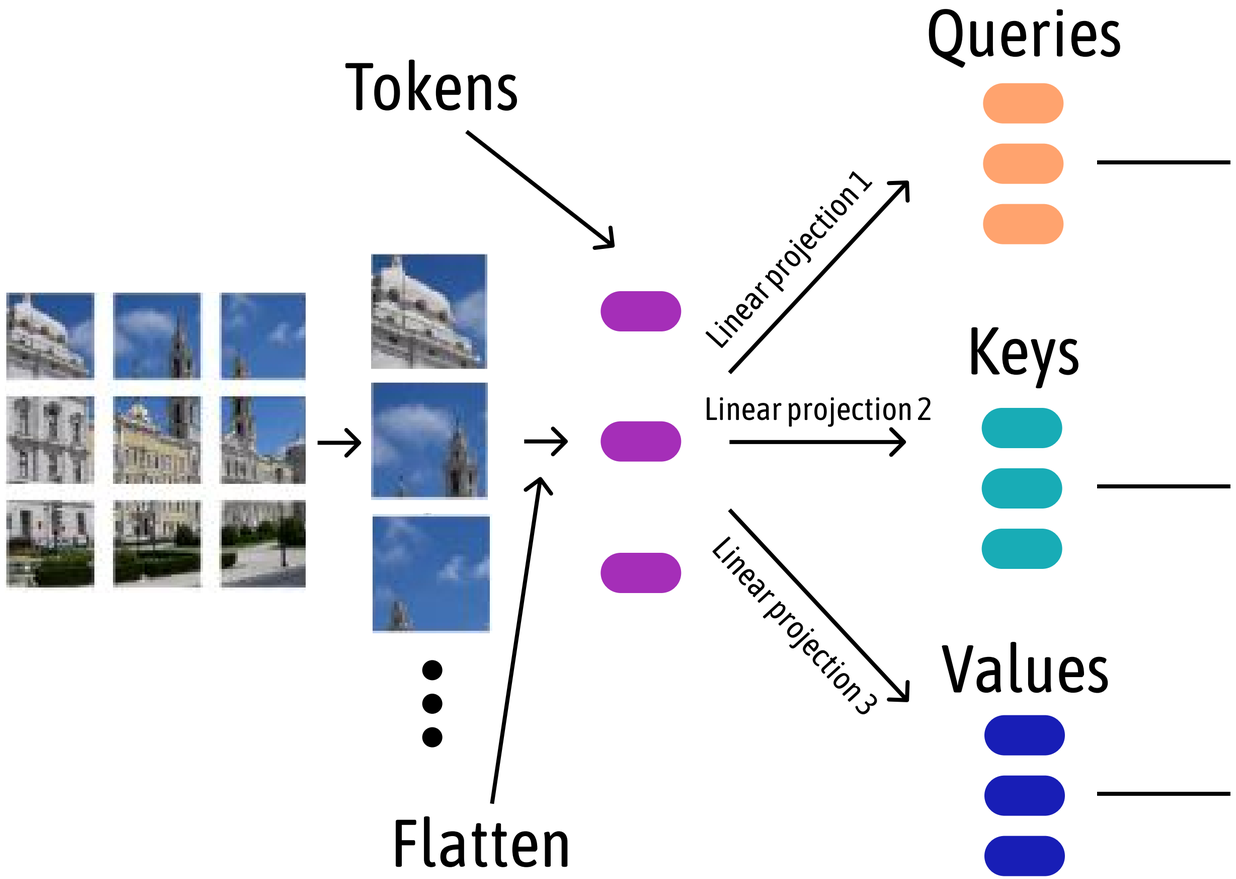
Final, computed pose/representation for the object

Estimated pose of the whole

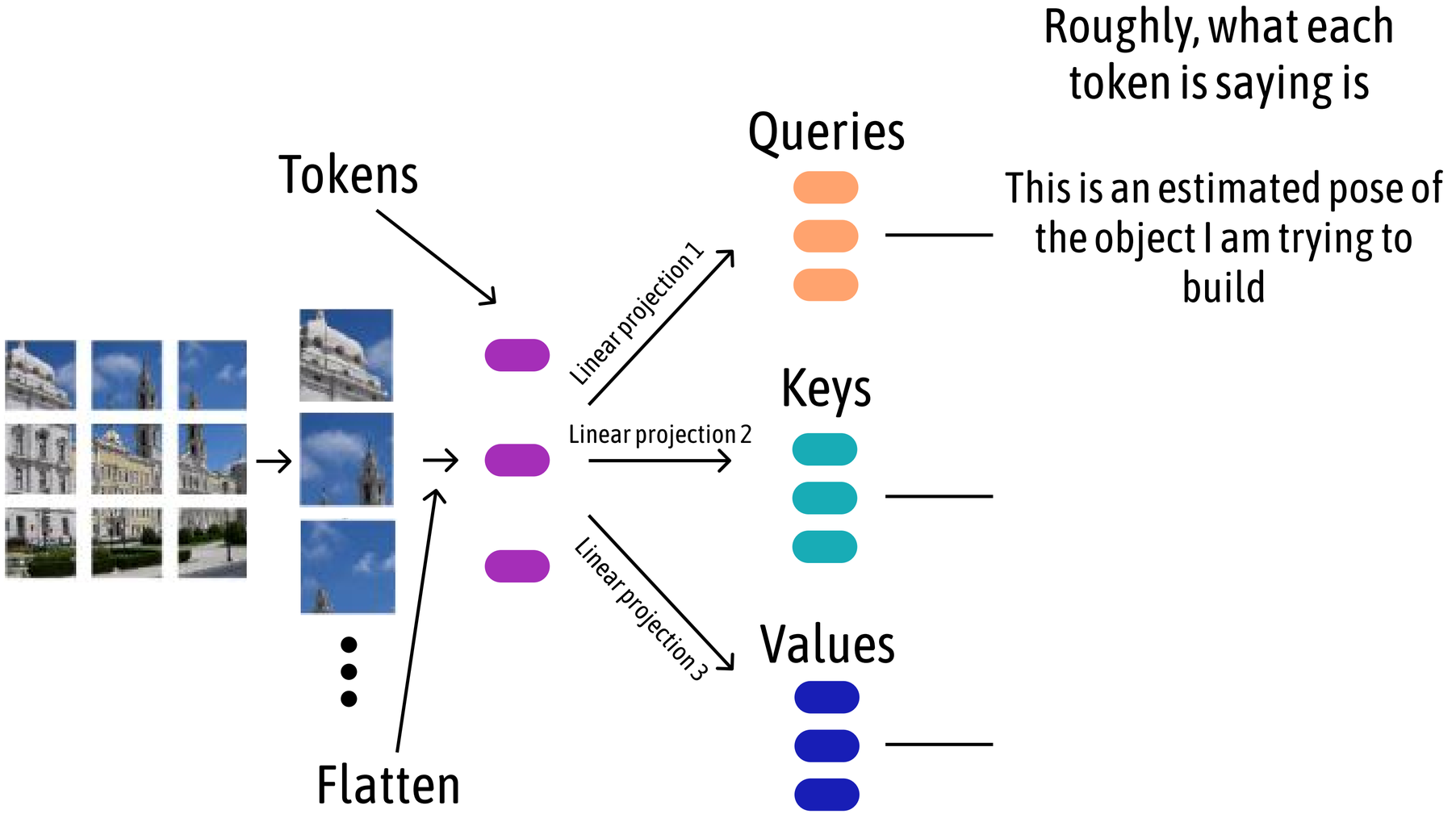
Routing by agreement

Core Computational Block - Self attention

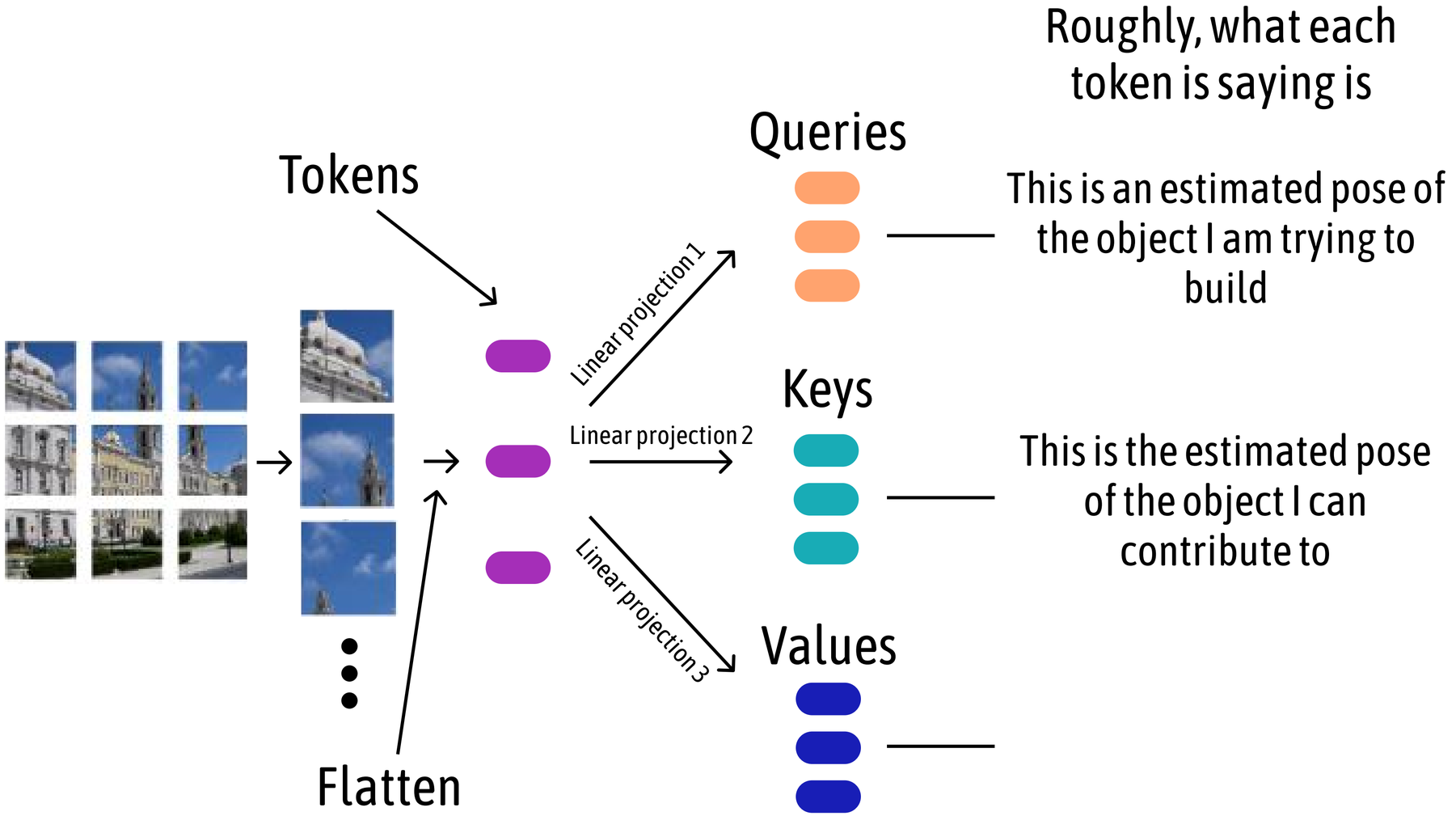
Roughly, what each token is saying is



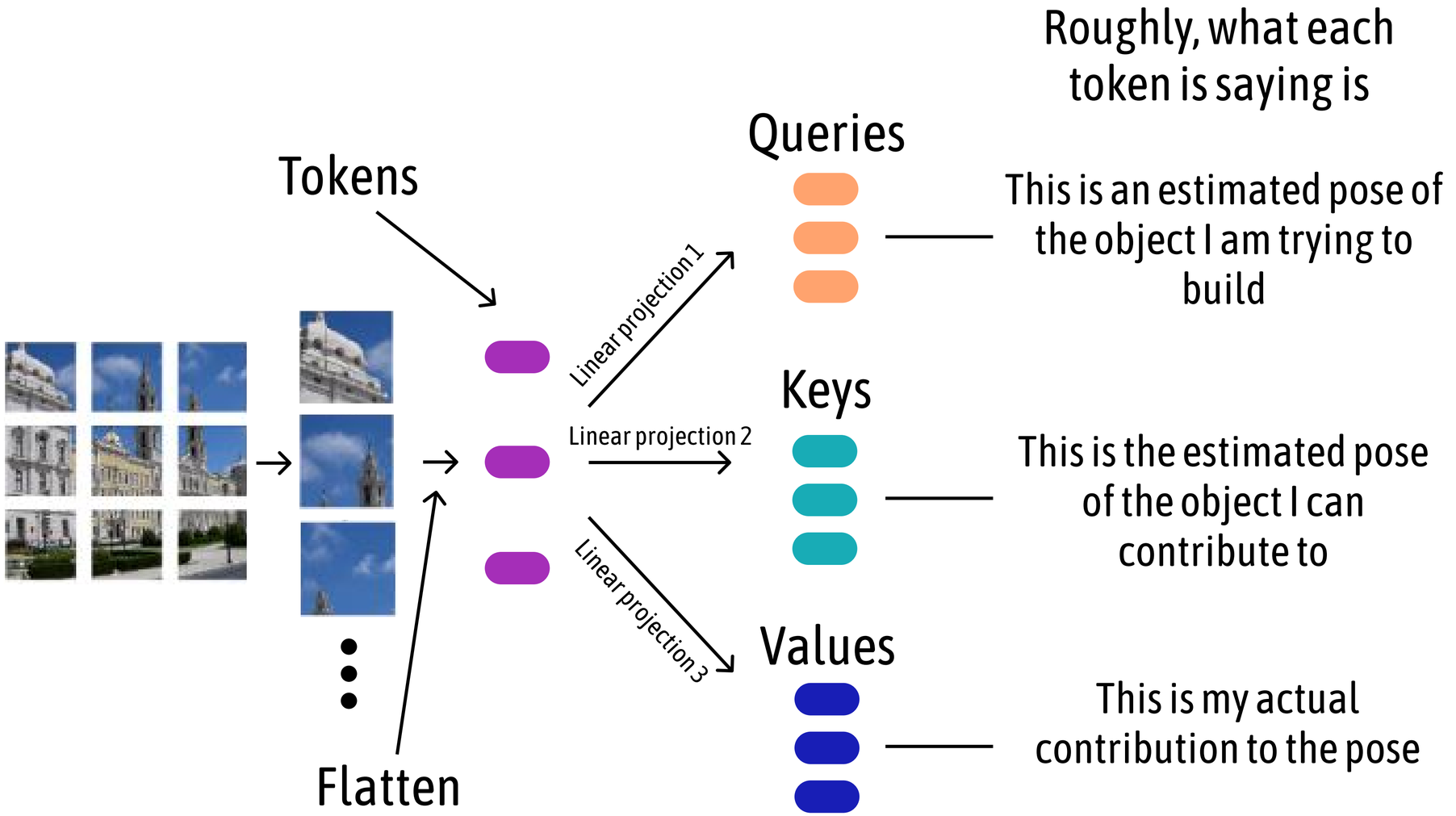
Core Computational Block - Self attention



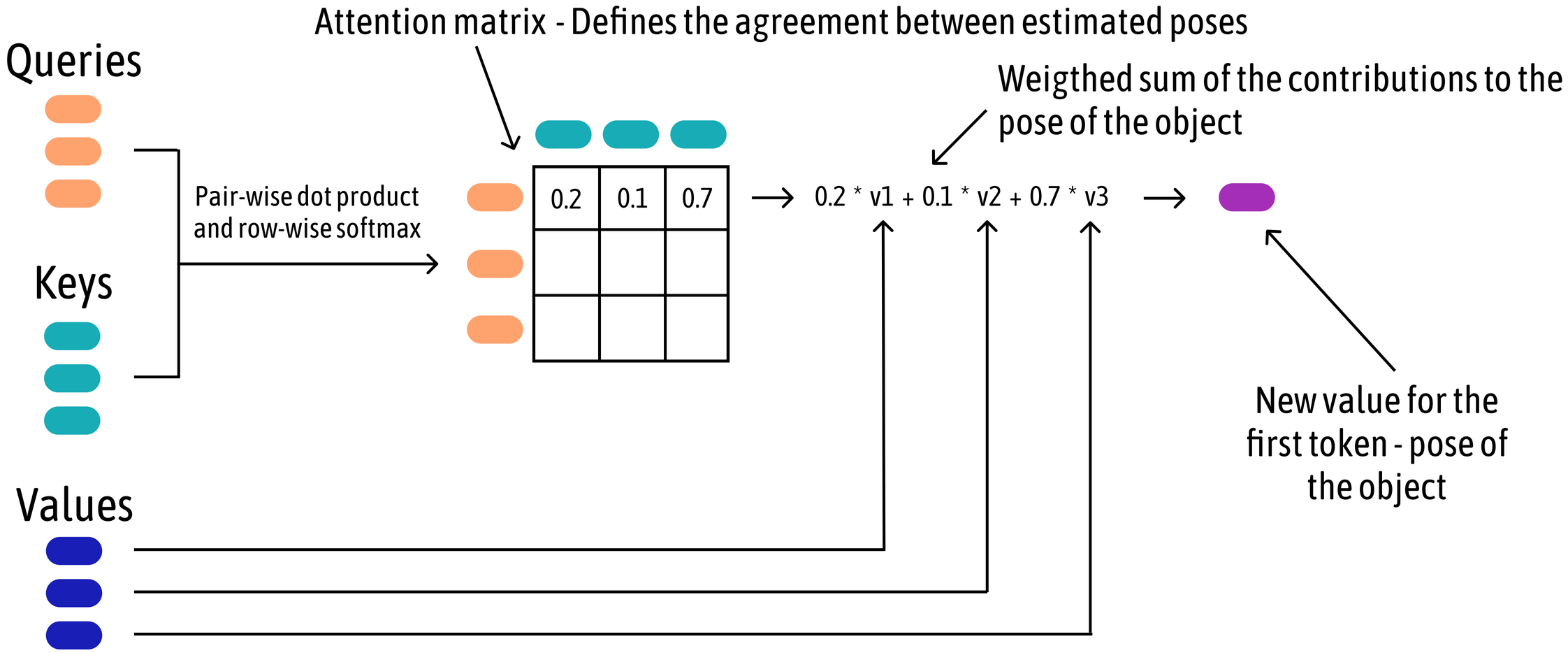
Core Computational Block - Self attention



Core Computational Block - Self attention

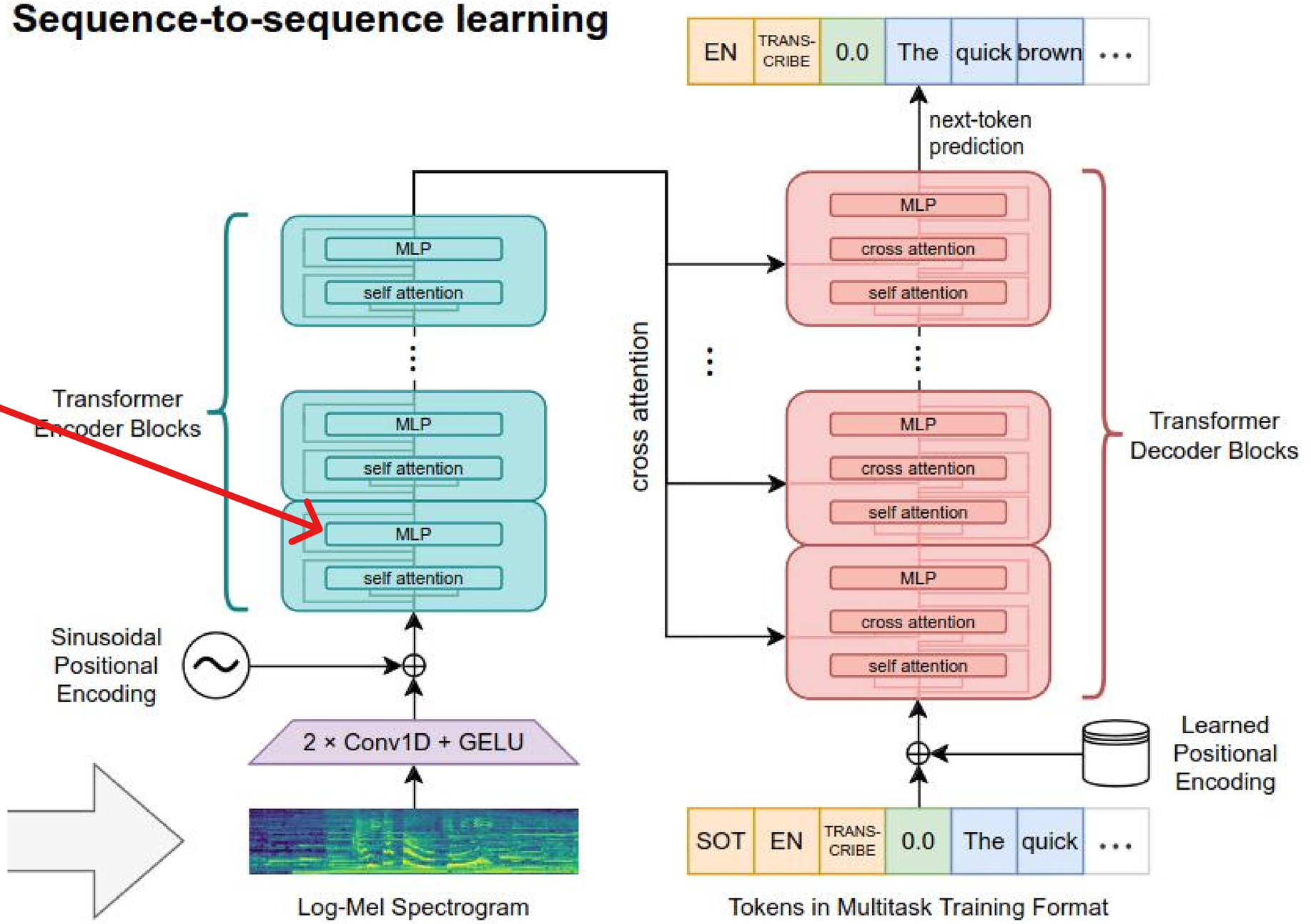


Core Computational Block - Self attention



Overview

Simple matrix multiplication for each individual token. It's the same matrix for all tokens.



Vocabulary and Model Output

Vocabulary - Start from text

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s. ¿Qué te gusta hacer en tu tiempo libre? كيف كان يومك اليوم؟ 今日はどんな一日でしたか？ Comment s'est passée ta journée aujourd'hui ? 你今天过得怎么样？איך היה היום שלך? Unjani usuku lwakho namhlanje? (Zulu for "How is your day today?") Bagaimana hari Anda hari ini?

Vocabulary - Convert text to bytes

Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value	Hex	Value
00	NUL	10	DLE	20	SP	30	0	40	@	50	P	60	`	70	p
01	SOH	11	DC1	21	!	31	1	41	A	51	Q	61	a	71	q
02	STX	12	DC2	22	"	32	2	42	B	52	R	62	b	72	r
03	ETX	13	DC3	23	#	33	3	43	C	53	S	63	c	73	s
04	EOT	14	DC4	24	$	34	4	44	D	54	T	64	d	74	t
05	ENQ	15	NAK	25	%	35	5	45	E	55	U	65	e	75	u
06	ACK	16	SYN	26	&	36	6	46	F	56	V	66	f	76	v
07	BEL	17	ETB	27	'	37	7	47	G	57	W	67	g	77	w
08	BS	18	CAN	28	(	38	8	48	H	58	X	68	h	78	x
09	HT	19	EM	29	)	39	9	49	I	59	Y	69	i	79	y
0A	LF	1A	SUB	2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
0B	VT	1B	ESC	2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
0C	FF	1C	FS	2C	,	3C	<	4C	L	5C	\	6C	l	7C	|
0D	CR	1D	GS	2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
0E	SO	1E	RS	2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
0F	SI	1F	US	2F	/	3F	?	4F	O	5F	_	6F	o	7F	DEL

Vocabulary - Convert text to bytes

128	Ç	144	É	160	á	176	░	193	┴	209	╤	225	ß	241	±
129	ü	145	æ	161	í	177	▒	194	┬	210	╥	226	Γ	242	≥
130	é	146	Æ	162	ó	178	▓	195	├	211	╙	227	π	243	≤
131	â	147	ô	163	ú	179	│	196	─	212	╘	228	Σ	244	⌠
132	ä	148	ö	164	ñ	180	┤	197	┼	213	╒	229	σ	245	⌡
133	à	149	ò	165	Ñ	181	╡	198	╞	214	╓	230	µ	246	÷
134	å	150	û	166	ª	182	╢	199	╟	215	╫	231	τ	247	≈
135	ç	151	ù	167	º	183	╖	200	╚	216	╪	232	Φ	248	°
136	ê	152	ÿ	168	¿	184	╕	201	╔	217	┘	233	Θ	249	∙
137	ë	153	Ö	169	⌐	185	╣	202	╩	218	┌	234	Ω	250	·
138	è	154	Ü	170	¬	186	║	203	╦	219	█	235	δ	251	√
139	ï	156	£	171	½	187	╗	204	╠	220	▄	236	∞	252	ⁿ
140	î	157	¥	172	¼	188	╝	205	═	221	▌	237	φ	253	²
141	ì	158	₧	173	¡	189	╜	206	╬	222	▐	238	ε	254	■
142	Ä	159	ƒ	174	«	190	╛	207	╧	223	▀	239	∩	255	
143	Å	192	└	175	»	191	┐	208	╨	224	α	240	≡		

Vocabulary - Convert text to bytes

133 201 23 12 77 155 152 88 173 35 23 12 183 102 169
72 230 209 84 49 70 3 49 160 38 91 145 240 224 52
141 228 203 247 105 175 23 12 237 52 147 148 203 10
67 156 77 189 203 23 12 195 173 170 188 20 57 177
53 142 80 120 75 167 168 192 239 23 12 134 135 49 70
7 87 165 115 75 142 32 206 100 191 45 150 29 132
196 44 65 34 151 143 23 12 152 54 24 0 74 200 23
12 158 38 164 151 49 166 97 24 23 12 144 97 192

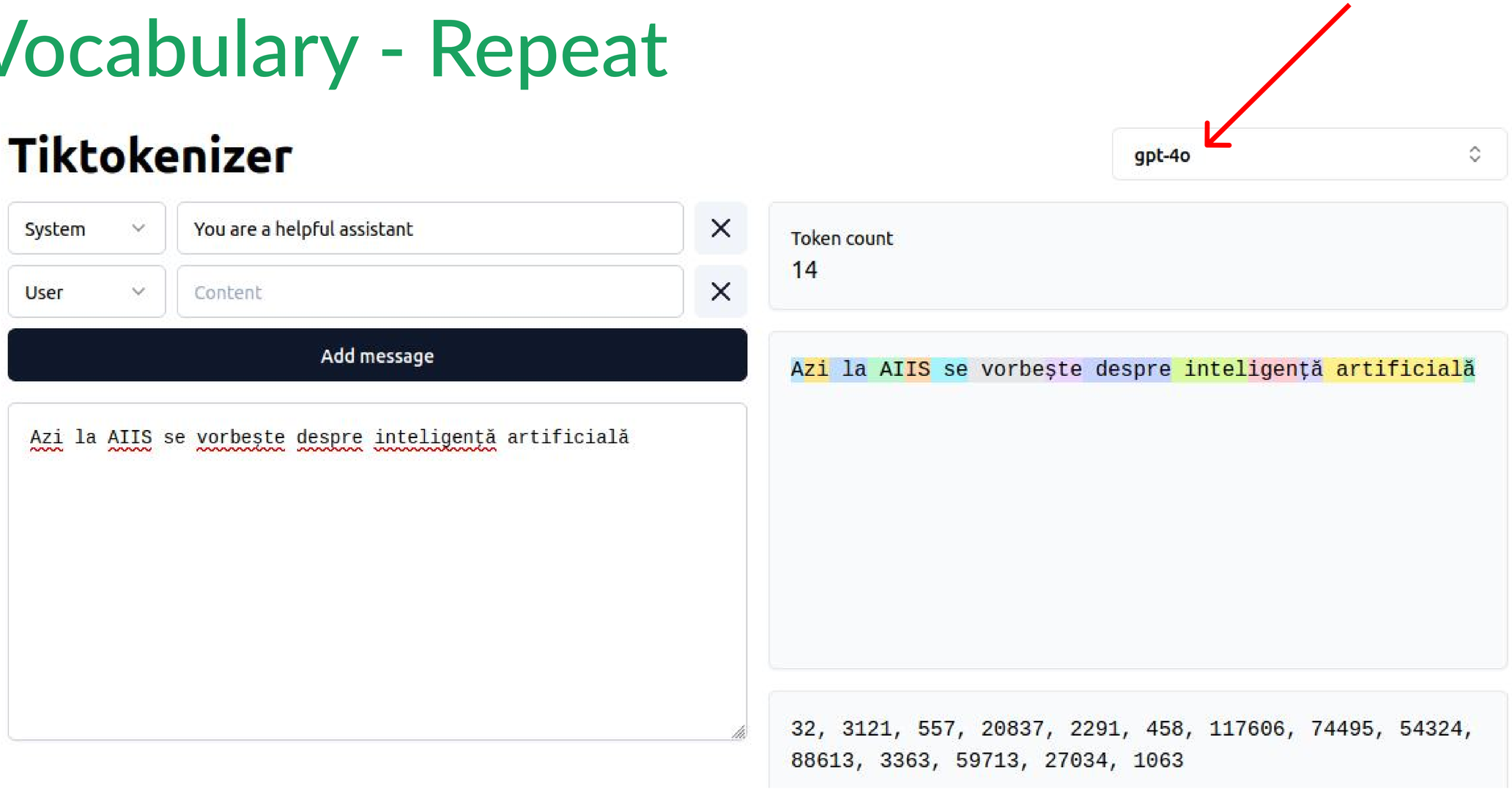
Vocabulary - Find the most frequent pair

133 201 23 12 77 155 152 88 173 35 23 12 183 102 169
72 230 209 84 49 70 3 49 160 38 91 145 240 224 52
141 228 203 247 105 175 23 12 237 52 147 148 203 10
67 156 77 189 203 23 12 195 173 170 188 20 57 177
53 142 80 120 75 167 168 192 239 23 12 134 135 49 70
7 87 165 115 75 142 32 206 100 191 45 150 29 132
196 44 65 34 151 143 23 12 152 54 24 0 74 200 23
12 158 38 164 151 49 166 97 24 23 12 144 97 192

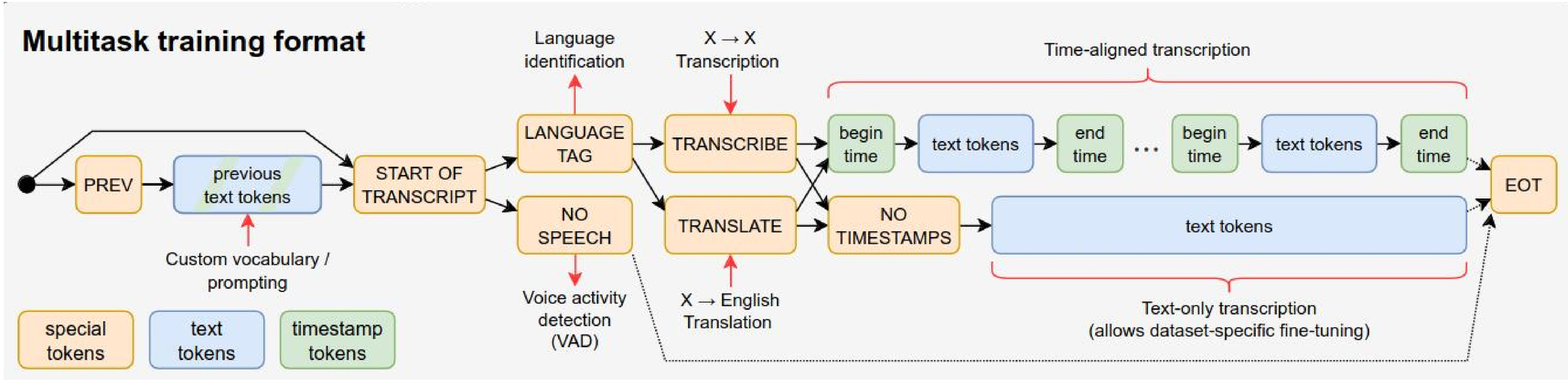
Vocabulary - Replace the pair (new entry)

133 201 256 77 155 152 88 173 35 256 183 102 169 72
230 209 84 49 70 3 49 160 38 91 145 240 224 52 141
228 203 247 105 175 256 237 52 147 148 203 10 67
156 77 189 203 256 195 173 170 188 20 57 177 53 142
80 120 75 167 168 192 239 23 12 134 135 49 70 7 87
165 115 75 142 32 206 100 191 45 150 29 132 196 44
65 34 151 143 256 152 54 24 0 74 200 256 158 38
164 151 49 166 97 24 256 144 97 192

Vocabulary - Repeat



Output Format



Other Resources

Intro (+implementation) to Deep Learning:

https://www.youtube.com/watch?v=Y6YcK7Lhdd4

Intro (+implementation) to Transformers:

https://www.youtube.com/watch?v=kCc8FmEb1nY

Intro (+implementation) to Tokenizers:

https://www.youtube.com/watch?v=zduSFxRajkE

Demo

Thank You