Neural Networks meet Explicit Knowledge Representation:

Towards Mechanistic Interpretability and Neuro-symbolic Modeling by-design

Elena Sofia Ruzzetti[*], Leonardo Ranaldi[*,o], Fabio Massimo Zanzotto[*,+]

[*]Human Centric ART, University of Rome Tor Vergata

[o] University of Edinburgh

[+]AlmaWave

TOR VERGATA
UNIVERSITY OF ROME

Human Centric ART

ALMAWAVE

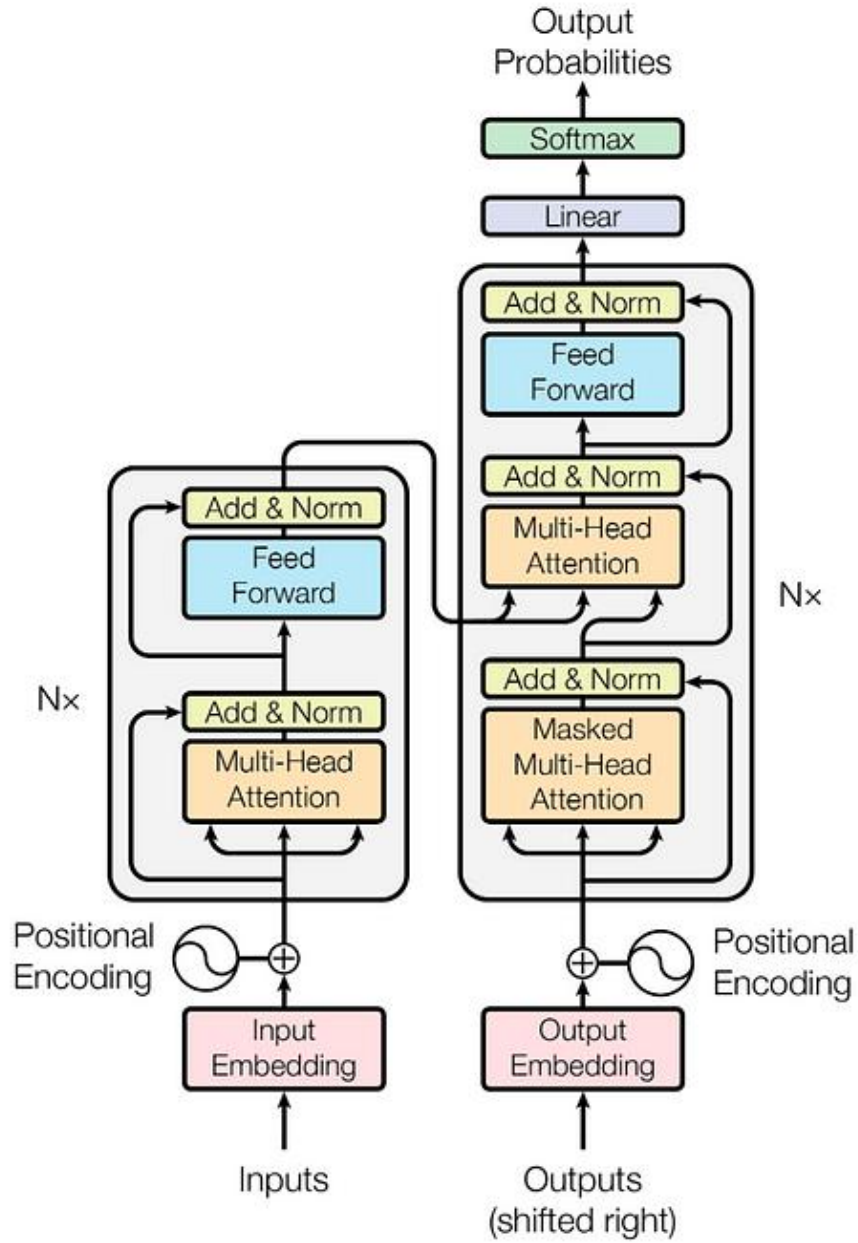


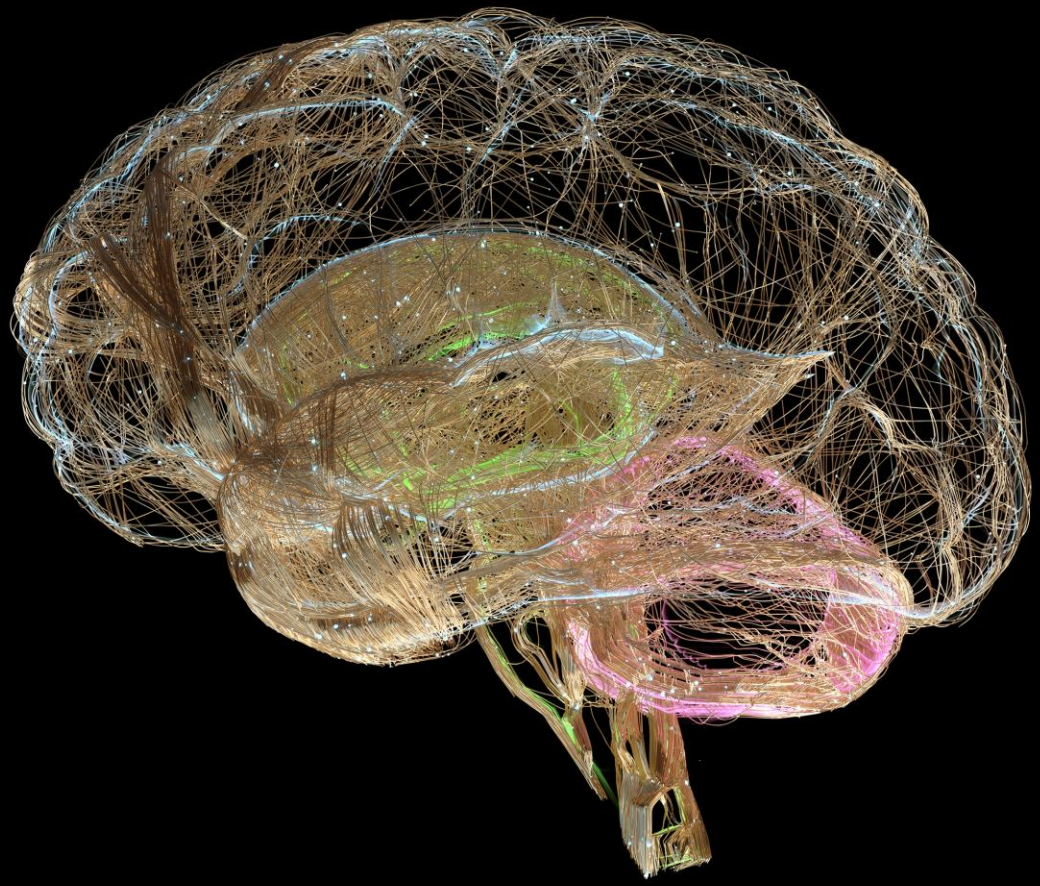
Figure 1: The Transformer - model architecture.

Mechanistic Interpretability or Neurosymbolic AI? This is the dilemma





NEURAL NETWORKS



ATTENTION IS ALL YOU NEED

The

MATRIX

can «think»

The multiplication among

MATRIX CES

can «think»

Reading what machines "think"

Authors:  Fabio Massimo Zanzotto, Danilo Croce

Authors Info & Claims

BI'09: Proceedings of the 2009 international conference on Brain informatics • Pages 159 - 170

The "multiplication among

MATRIXES

can «think»

Reading what machines "think"

Authors:  Fabio Massimo Zanzotto, Danilo Croce Authors Info & Claims

BI'09: Proceedings of the 2009 international conference on Brain informatics • Pages 159 - 170

The multiplication among

Parallels between machine and brain decoding

Authors: Lorenzo Dell'Arciprete, Brian Murphy,  Fabio Massimo Zanzotto Authors Info & Claims

BI'12: Proceedings of the 2012 international conference on Brain Informatics • Pages 162 - 174
https://doi.org/10.1007/978-3-642-35139-6_16

can «think»

Reading what machines "think"

Authors:  Fabio Massimo Zanzotto, Danilo Croce | Authors Info & Claims

BI'09: Proceedings of the 2009 international conference on Brain informatics • Pages 159 - 170

The multiplication among

Parallels between machine and brain decoding

Authors: Lorenzo Dell'Arciprete, Brian Murphy,  Fabio Massimo Zanzotto | Authors Info & Claims

BI'12: Proceedings of the 2012 international conference on Brain Informatics • Pages 162 - 174
https://doi.org/10.1007/978-3-642-35139-6_16

can «think»

Using Distributional Models for Studying the Influence of School Textbooks in Children Bias

Publisher: IEEE Cite This PDF

Elena Sofia Ruzzetti ; Davide Venditti ; Fabio Massimo Zanzotto ; Francesca Fallucchi All Authors

Reading what machines "think"

Authors: Fabio Massimo Zanzotto, Danilo Croce | Authors Info & Claims

BI'09: Proceedings of the 2009 international conference on Brain informatics • Pages 159 - 170

Parallels between machine and brain decoding

Authors: Lorenzo Dell'Arciprete, Brian Murphy, Fabio Massimo Zanzotto | Authors Info & Claims

BI'12: Proceedings of the 2012 international conference on Brain Informatics • Pages 162 - 174
https://doi.org/10.1007/978-3-642-35139-6_16

among

MATRIX

YES

Using Distributional Models for Studying the Influence of School Textbooks in Children Bias

Publisher: IEEE Cite This PDF

Elena Sofia Ruzzetti ; Davide Venditti ; Fabio Massimo Zanzotto ; Francesca Fallucchi All Authors

can "think"

Animate, or Inanimate, That Is the Question for Large Language Models

by Giulia Pucci [1], Fabio Massimo Zanzotto [2] and Leonardo Ranaldi [1,2,*]

The multiplication among

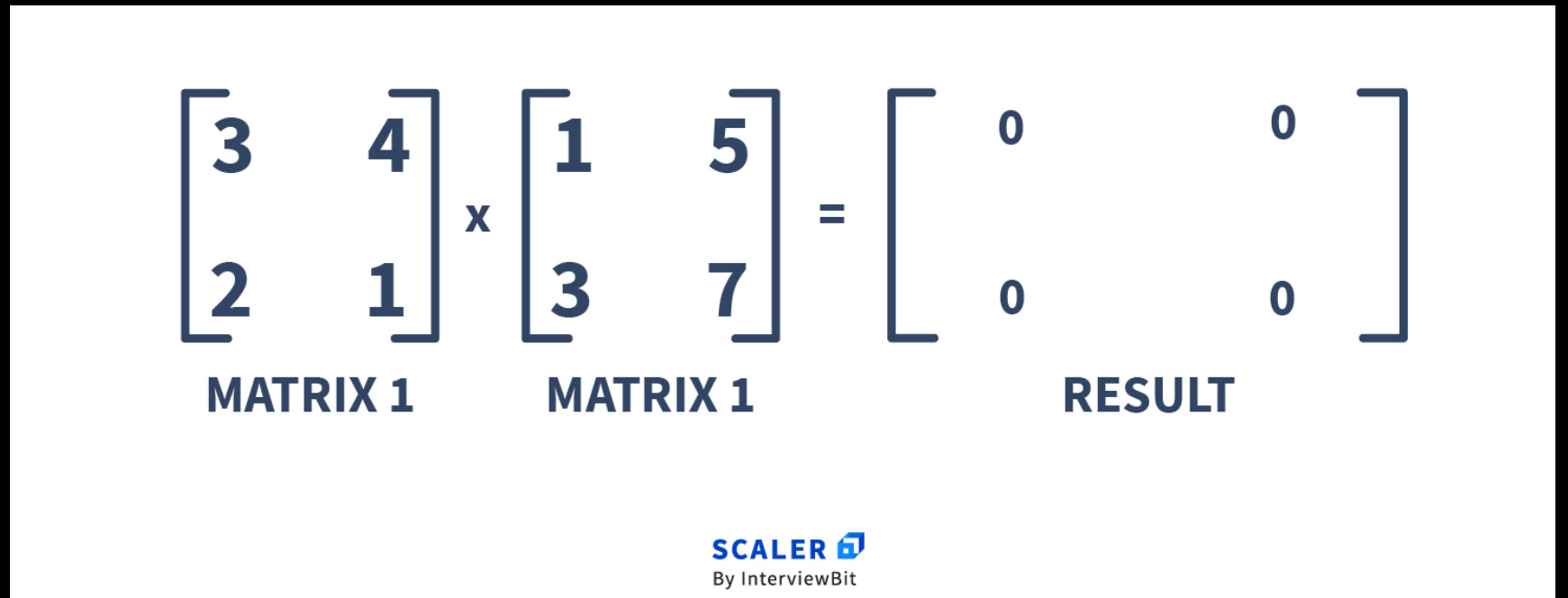
MATRIX CES

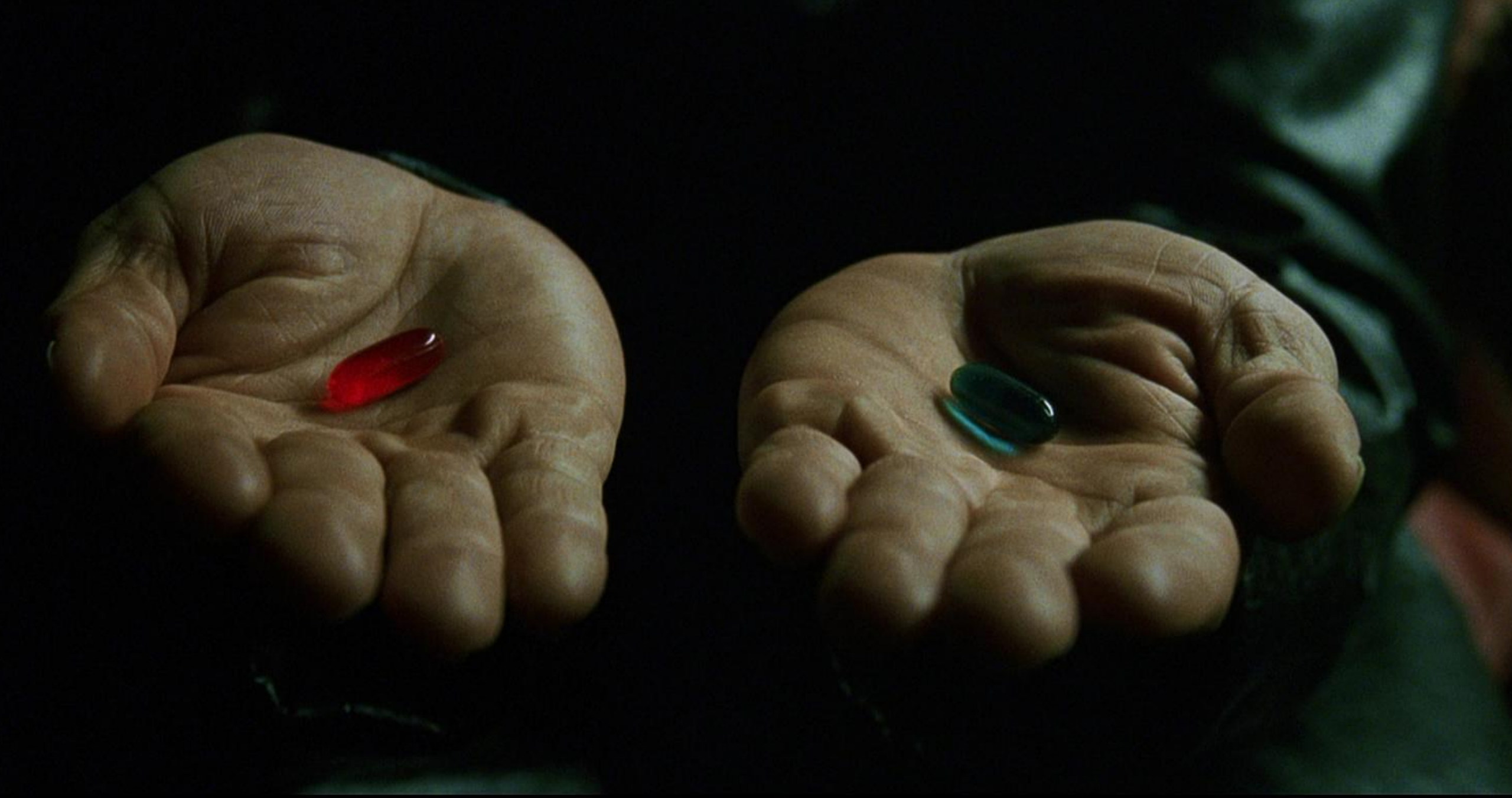
can «think»

The multiplication among

MATRIX~~X~~CES

can «think»











A Language Model

$w_1 \; w_2 \; \ldots\ldots \; w_n \; w_{n+1}$?

$$P(w_{n+1} \mid w_1\ w_2 \ldots\ldots\ w_n) =$$

$$= \frac{P(w_1\ w_2 \ldots\ldots\ w_n\ w_{n+1})}{P(w_1\ w_2 \ldots\ldots\ w_n)}$$

$$= \frac{P(w_1\, w_2 \ldots\ldots w_n w_{n+1})}{P(w_1\, w_2 \ldots\ldots w_n)} \triangleq$$

$$\frac{count(w_1\, w_2 \ldots\ldots w_n w_{n+1})}{Count(w_1\, w_2 \ldots\ldots w_n)}$$

Building the sample space

He enrolled in the mathematics and physics teaching diploma program.

([He, enrolled, in, the], **mathematics**)

([enrolled, in, the, mathematics], **and**)

([in, the, mathematics, and], **physics**)

([the, mathematics, and, physics], **teaching**)

([mathematics, and, physics, teaching], **diploma**)

([and, physics, teaching, diploma], **program**)

$$P(w_{n+1} \mid w_1\ w_2 \ldots\ldots\ w_n) =$$

$$= \frac{P(w_1\ w_2 \ldots\ldots\ w_n\ w_{n+1})}{P(w_1\ w_2 \ldots\ldots\ w_n)}$$

Finding probabilities for unseen sequences

Katz's back-off model

$$P_{bo}(w_i \mid w_{i-n+1} \cdots w_{i-1})$$

$$= \begin{cases} d_{w_{i-n+1} \cdots w_i} \dfrac{C(w_{i-n+1} \cdots w_{i-1} w_i)}{C(w_{i-n+1} \cdots w_{i-1})} & \text{if } C(w_{i-n+1} \cdots w_i) > k \\ \alpha_{w_{i-n+1} \cdots w_{i-1}} P_{bo}(w_i \mid w_{i-n+2} \cdots w_{i-1}) & \text{otherwise} \end{cases}$$

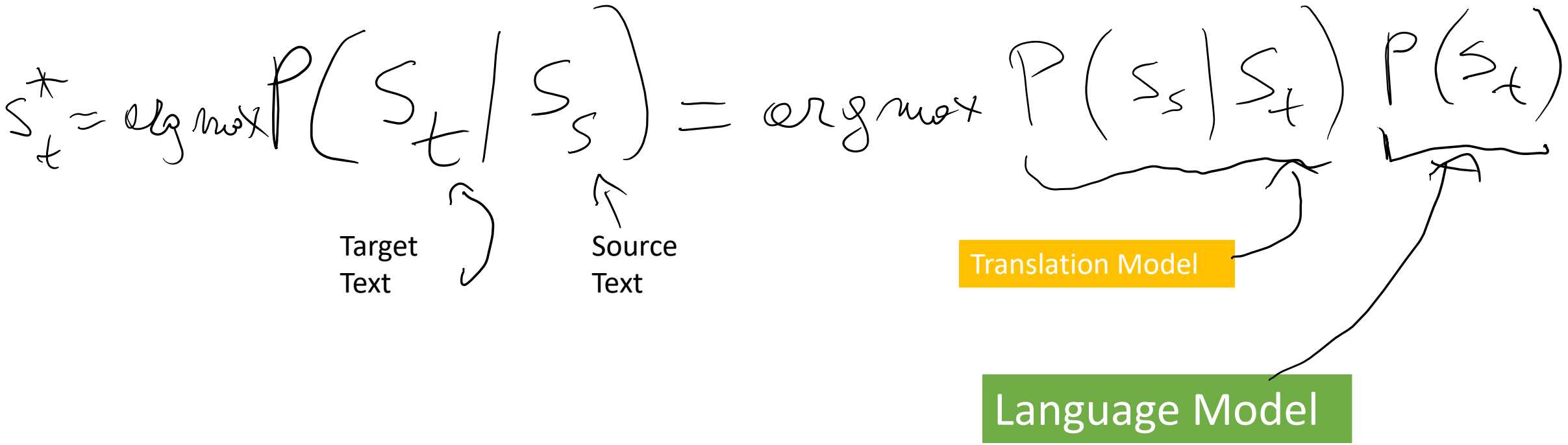
where

$C(x)$ = number of times x appears in training

w_i = ith word in the given context

Where have we seen a Language Model before?

Statistical Machine Translation



Are Distributed Representations in Neural Networks adding something mysterious to knowledge representation?

The classical debate

Position One

Distributed representations in Neural Network architectures are:

"only an implementation of the Classical approach"

classical approach is related to discrete symbolic representations (Fodor and Pylyshyn, 1988)

Position Two

distributed representations give the important opportunity to reason *"holistically"* about encoded knowledge (Chalmers, 1992)

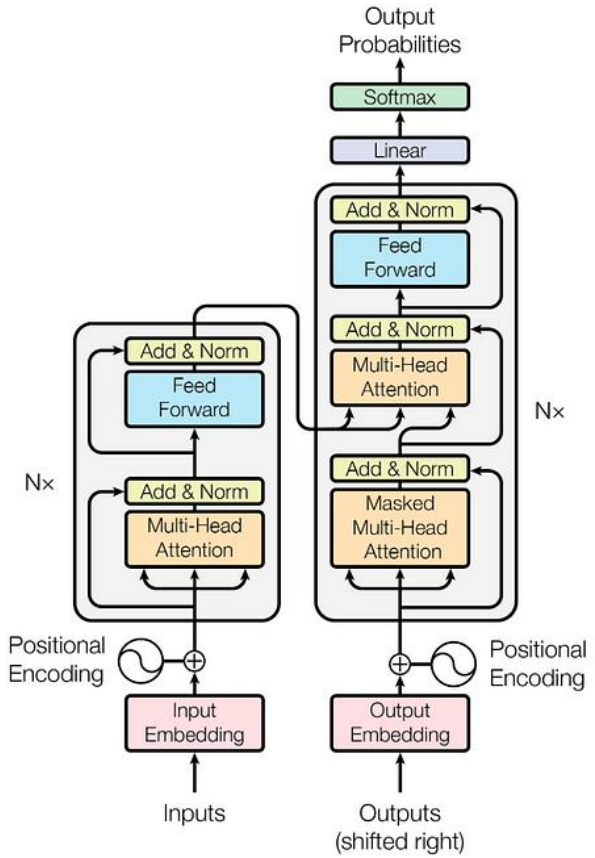


Figure 1: The Transformer - model architecture.

#UnboxingTransformers

Are Distributed Representations in Neural Networks adding something mysterious to knowledge representation?

The classical debate

Position One

Distributed representations in Neural Network architectures are:

"only an implementation of the Classical approach"

classical approach is related to discrete symbolic representations (Fodor and Pylyshyn, 1988)

Position Two

distributed representations give the important opportunity to reason *"holistically"* about encoded knowledge (Chalmers, 1992)

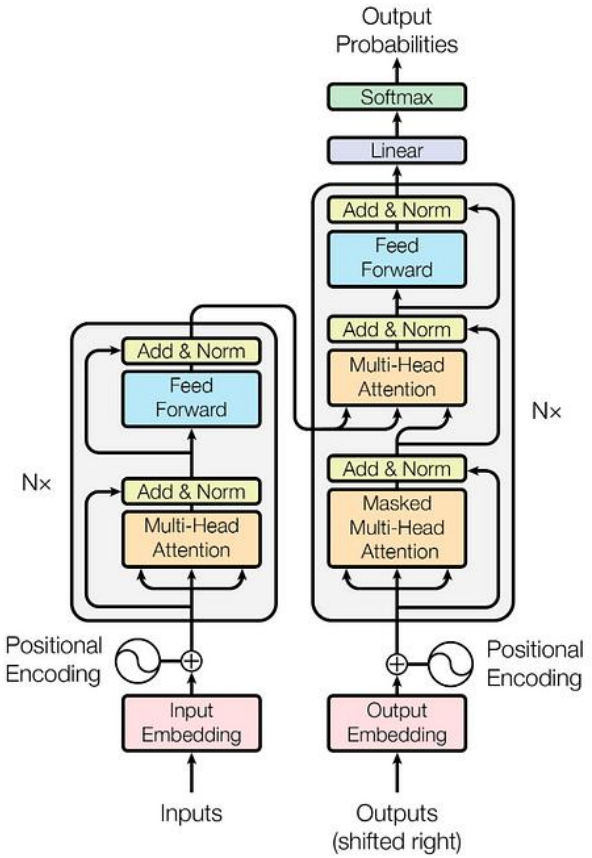
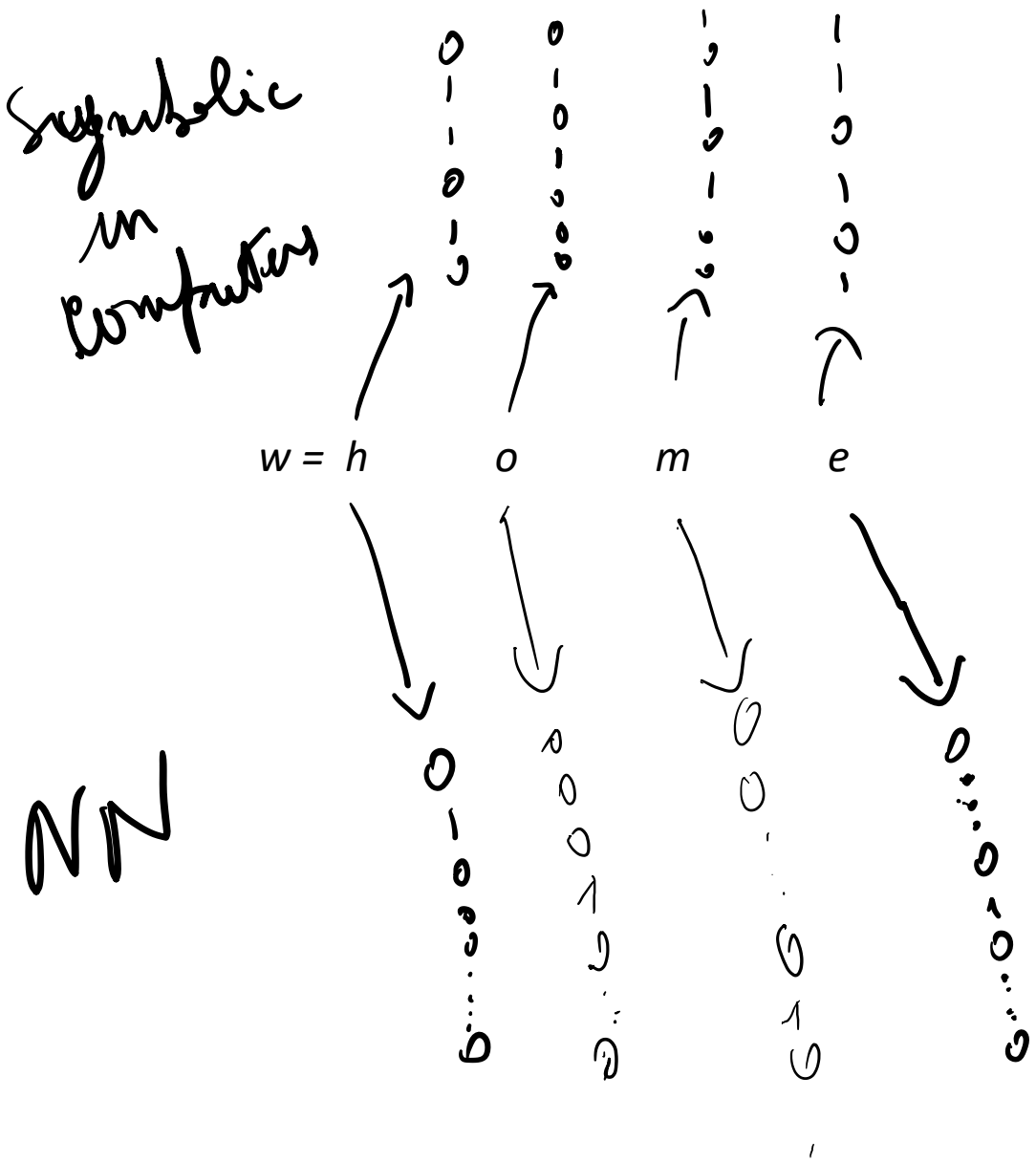


Figure 1: The Transformer - model architecture.

Representing Discrete Symbols with Vectors



bit representation

one hot representation

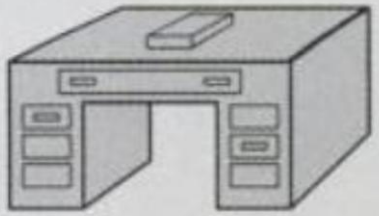
aka local representations (Plate, 1990)

A parallel with External Knowledge Representations in Cognitive Psycology

Traditional

Discrete Symbolic Representations

FIGURE 9.3

Language	Picture
"The book is on the desk"	
1. Discrete symbols	1. No discrete symbols
2. Explicit, needs symbol for relation	2. Implicit, no separate symbol for relation
3. Grammatical, clear rules of combination for types of symbol	3. No clear rules of combination or symbol types
4. Abstract	4. Concrete

Distributed

Local&Distributed Representations

Picture from: Michael W. Eysenck, Mark T. Keane, Cognitive Psychology: A Student's Handbook, 2000

Discrete Symbolic Representations: Compositionality

(Semantic) Compositionality
The meaning of a sequence of symbols can be inferred from the meaning of the symbols

Compositionality

(Gelder, 1990)

Functional compositionality

$$f(w_1, w_2, \ldots, w_2) = m$$

Concatenative compositionality

$$w_1 \circ w_2 \circ \ldots \circ w_n$$

The trivial property →

symbols can be recognized

Main Question

How much concatenative is functional compositionality?

Subsequent questions:

- Can **individual symbols** be recognized in the result of a functional compositionality function?
- Can the sequence of these individual symbols be reconstructed?

Firstly, Let's focus on linear functions

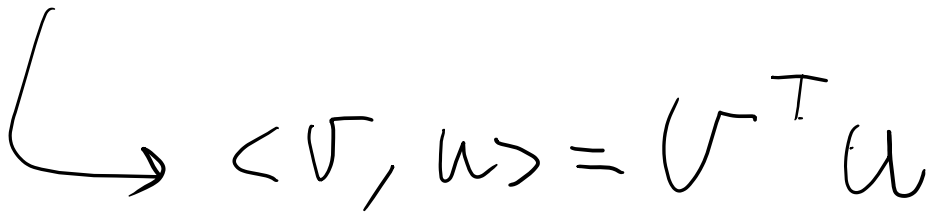
- Why should be interested in this concatentive property?

How to explore the problem

- Selecting a linear function

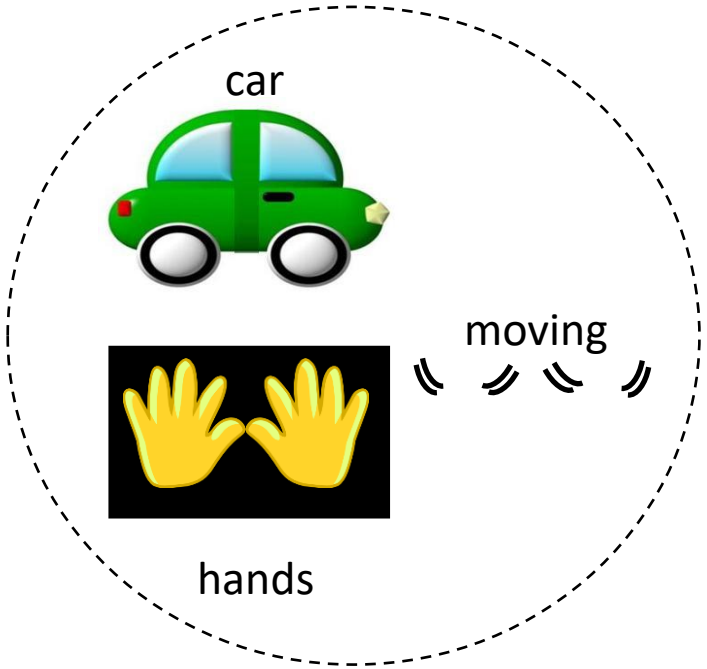
- Verifying the properties in the metric space:

  dot product is used to determine what happens in the final metric space after the application of the



Exploring composition functions

An embedding representing «word meaning»



A functional composition (sum), which can be possibly concatenative (?)

moving car

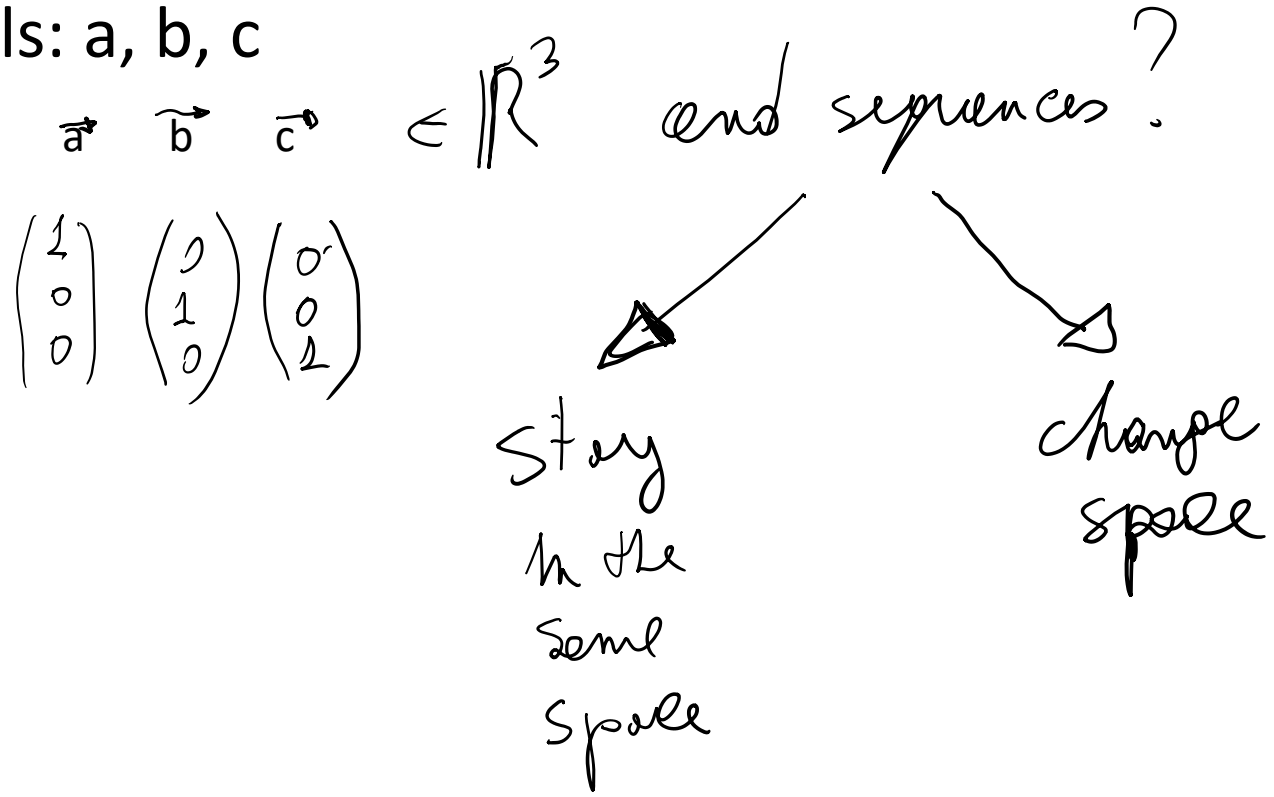


moving hands



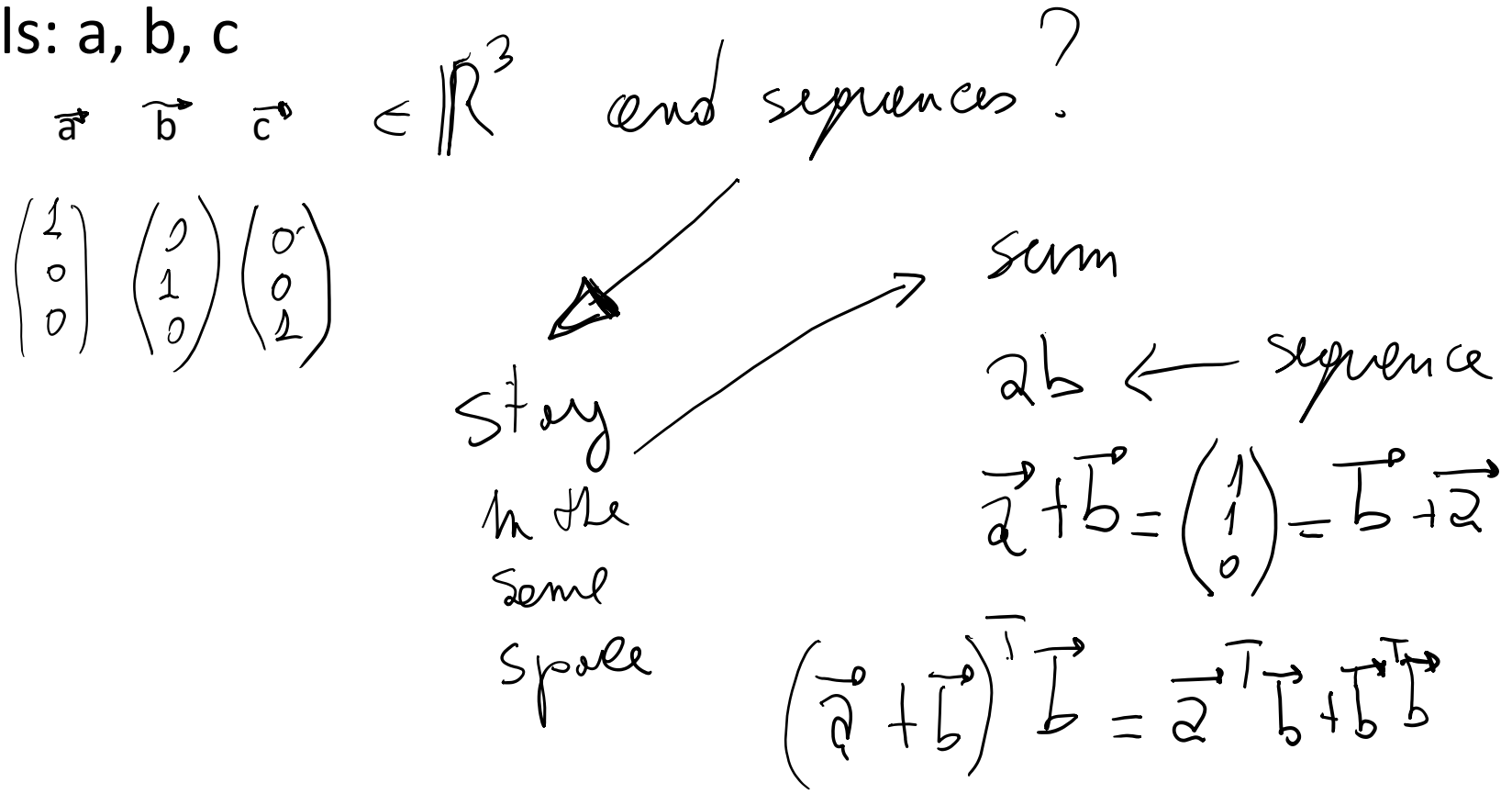
A very simple example with one-hot vectors

Three symbols: a, b, c



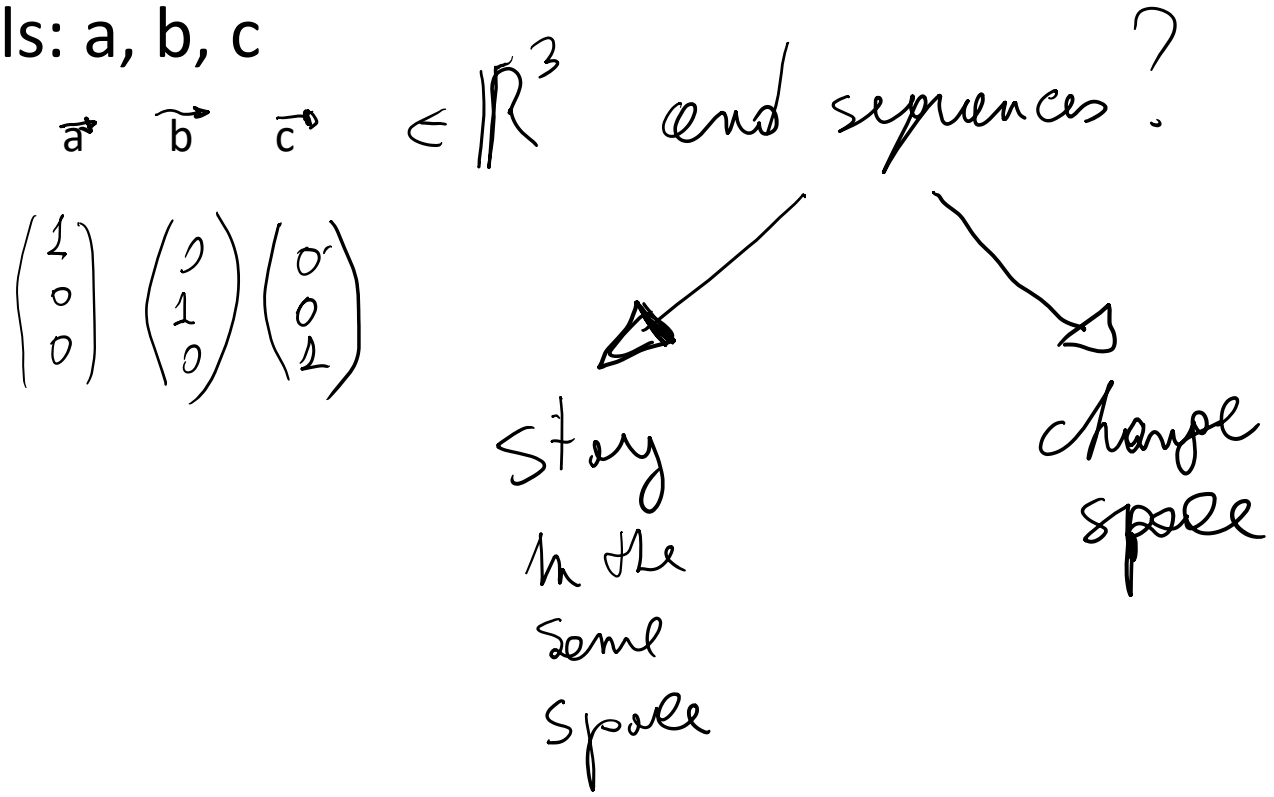
A very simple example with one-hot vectors

Three symbols: a, b, c



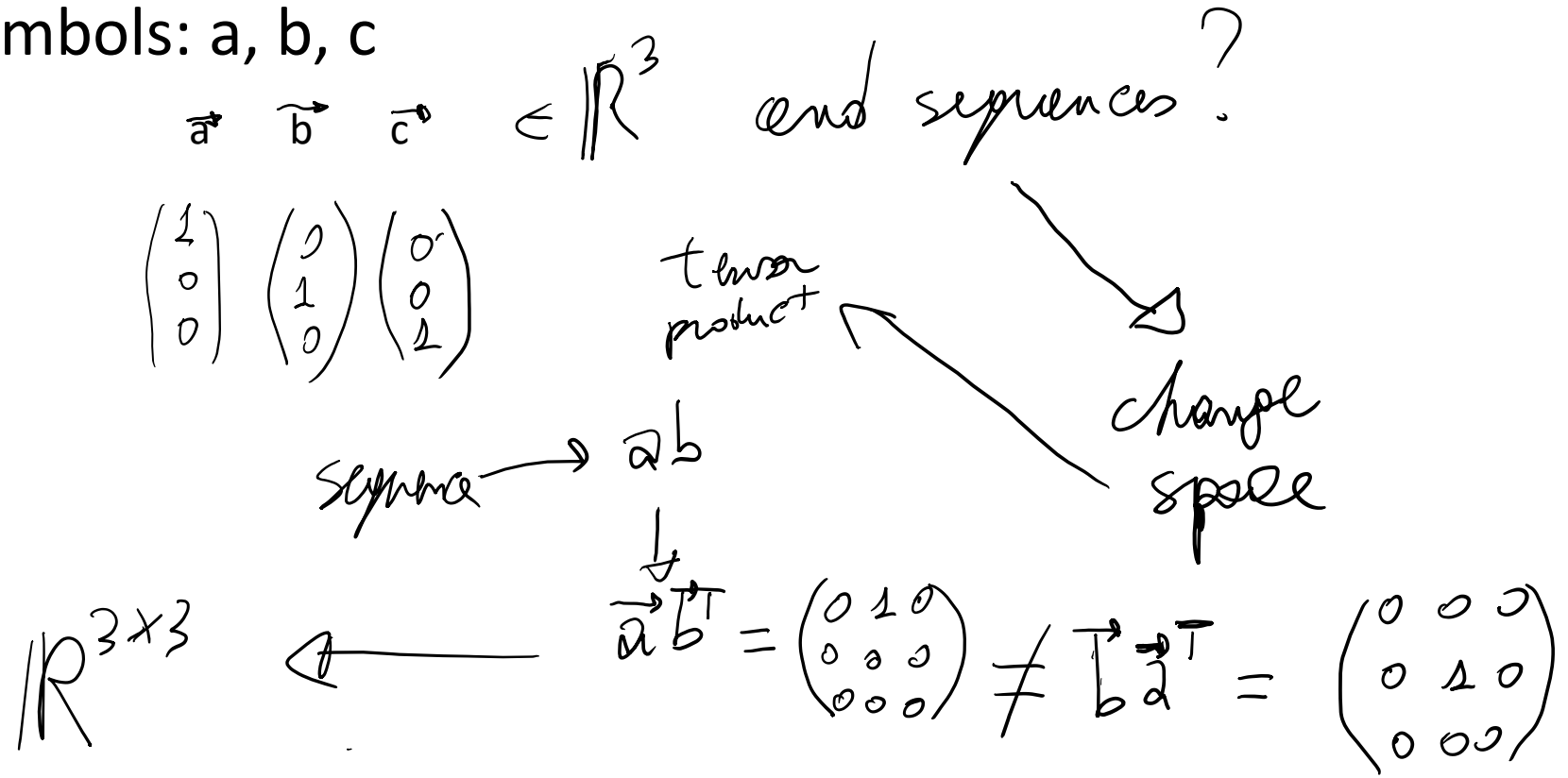
A very simple example with one-hot vectors

Three symbols: a, b, c



A very simple example with one-hot vectors

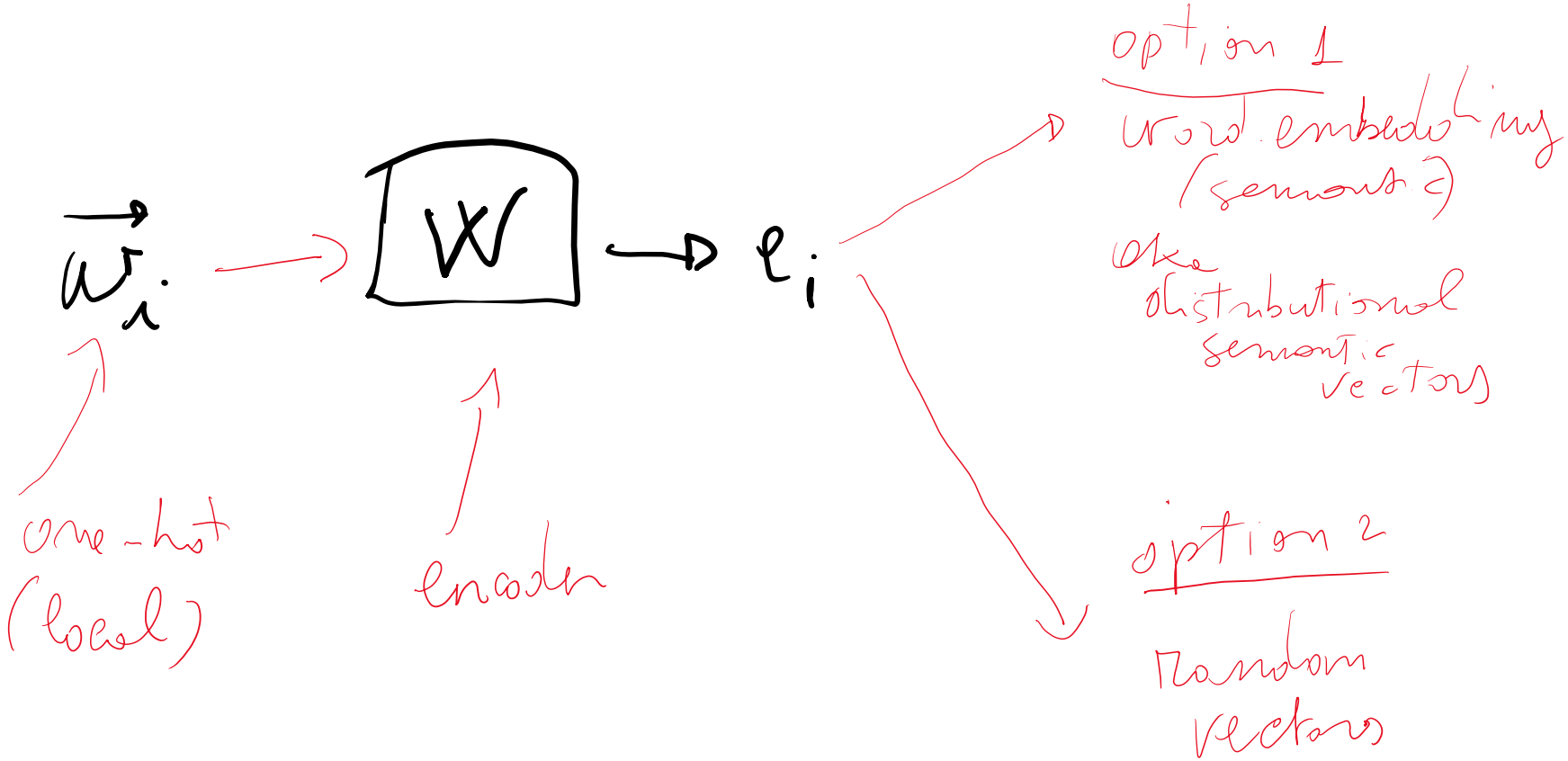
Three symbols: a, b, c



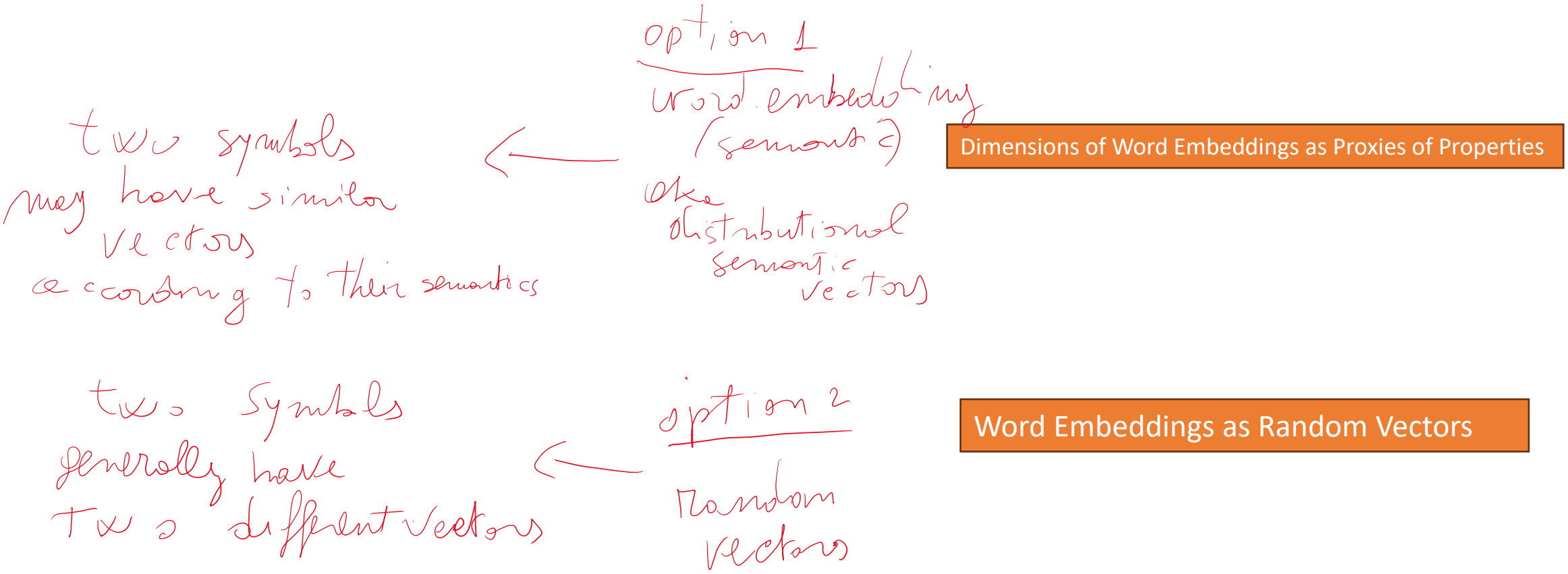
A very simple example with one-hot vectors

Can we embed the one-hot vectors in a smaller space?

What if we use symbol (word) embeddings



What if we use symbol (word) embeddings



Dimensions of Word Embeddings as Proxies of Properties

Word Embeddings as Random Vectors

Dimensions of Word Embeddings as Proxies of Properties

		the	sofa	point	and	table	picked	up	threw	it	
chair = (	…	3	1	0	2	1	1	1	1	1	…)

Option 1

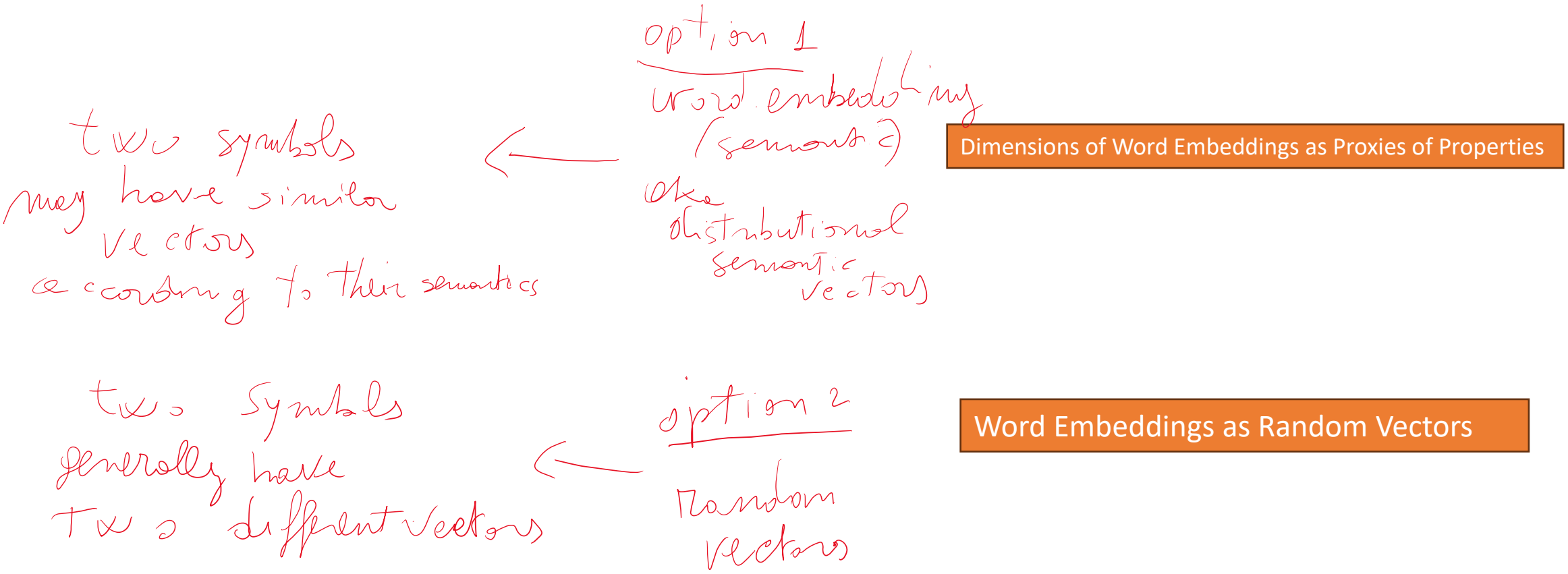
SURFACE: chair

SYNTATIC STRUCTURE: …

QUALIA STRUCTURE:

- FORMAL: seat
- CONSTITUTIVE : has_back, has_for_legs
- TELIC : sitting
- AGENTIVE:

What if we use symbol (word) embeddings



Dimensions of Word Embeddings as Proxies of Properties

Word Embeddings as Random Vectors

Word Embeddings as Random Vectors

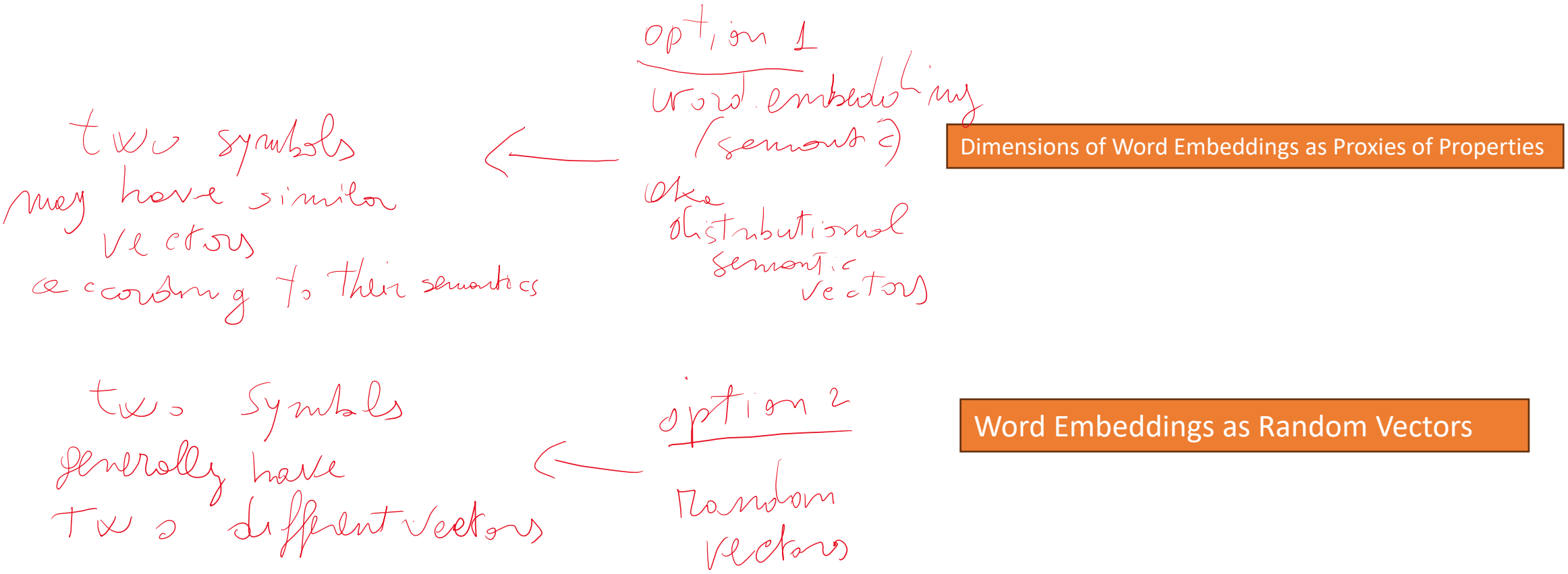
Vectors for symbols approximately keep the properties of one-hot vectors

$$\vec{a}^{\top}\vec{b} \approx \begin{cases} 0 & a \neq b \\ 1 & a = b \end{cases}$$

↓

Linear composition may be concatenative!

What if we use symbol (word) embeddings



Dimensions of Word Embeddings as Proxies of Properties

Word Embeddings as Random Vectors

Exploring **attention**

#UnboxingTransformers

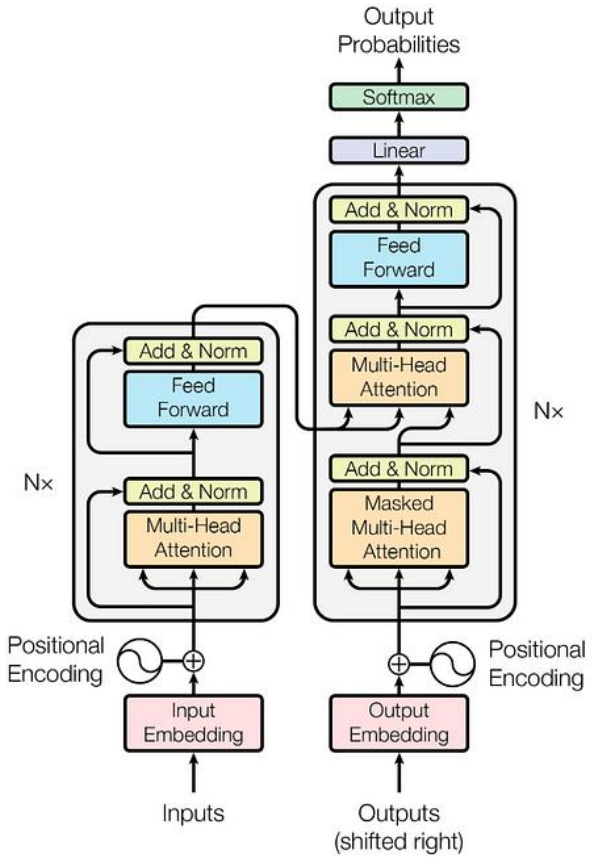


Figure 1: The Transformer - model architecture.

Exploring **attention** #UnboxingTransformers

Are Distributed Representations in Neural Networks adding something mysterious to knowledge representation?

The classical debate

Position One

Distributed representations in Neural Network architectures are:

"only an implementation of the Classical approach"

classical approach is related to discrete symbolic representations (Fodor and Pylyshyn, 1988)

Position Two

distributed representations give the important opportunity to reason *"holistically"* about encoded knowledge (Chalmers, 1992)

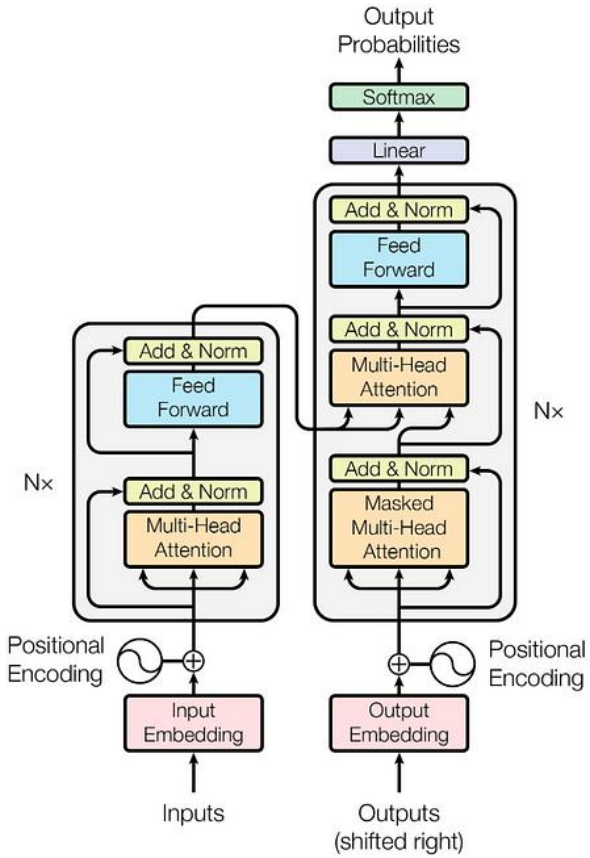


Figure 1: The Transformer - model architecture.

Exploring **attention** #UnboxingTransformers

Are Distributed Representations in Neural Networks adding someth

Position One

Distributed represe
architectures are:

"only an implem

classical approach i
representations (Fc

Position Two

distributed represe
opportunity to reas
knowledge (Chalme

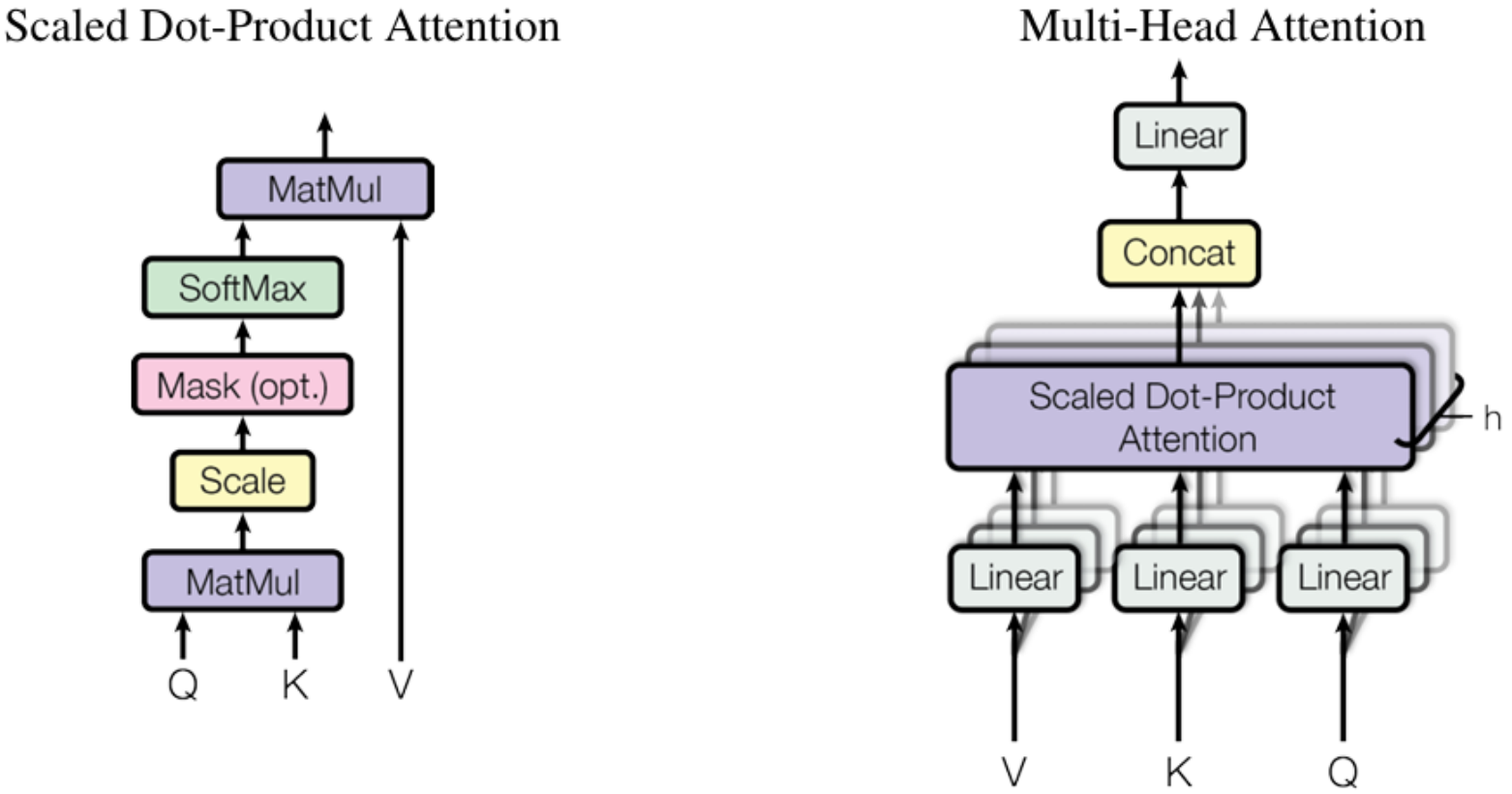


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

Expl

Are

som

Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

3.2.1 Scaled Dot-Product Attention

We call our particular attention "Scaled Dot-Product Attention" (Figure 2). The input consists of queries and keys of dimension d_k, and values of dimension d_v. We compute the dot products of the query with all keys, divide each by $\sqrt{d_k}$, and apply a softmax function to obtain the weights on the values.

In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V. We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

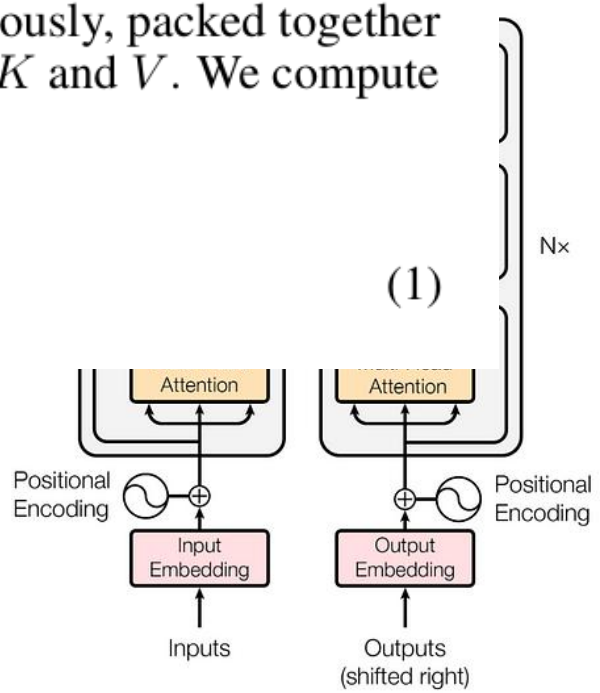


Figure 1: The Transformer - model architecture.

Position O

Distributed architectu

"only a

classical ap representa

Position T

distributed opportuni knowledge

Exploring **attention** #UnboxingTransformers

Are Distributed Representations in Neural Networks adding something mysterious to knowledge representation?

Position One

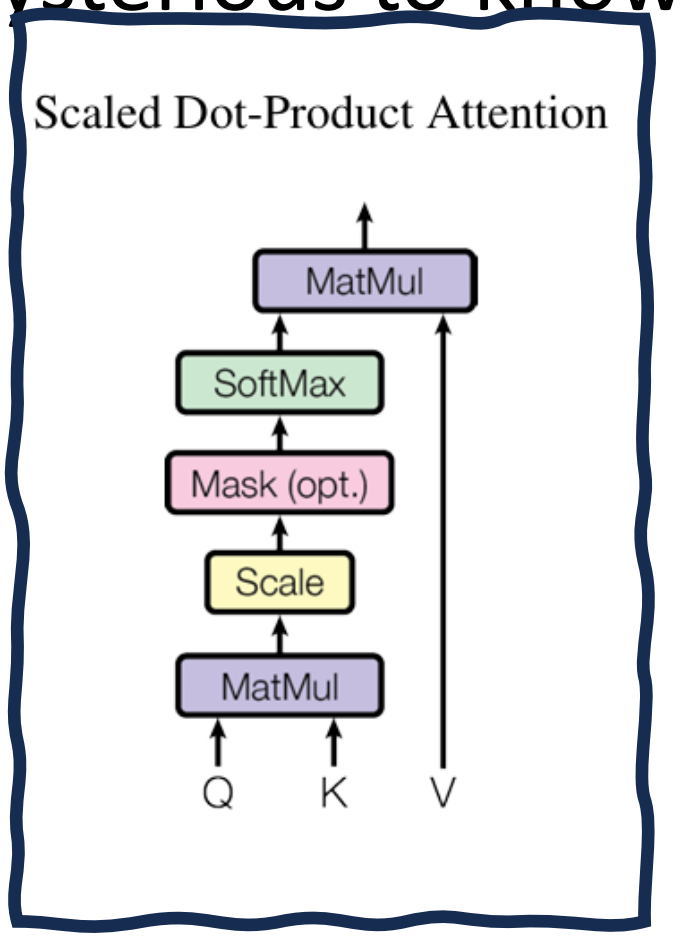
Distributed re
architectures

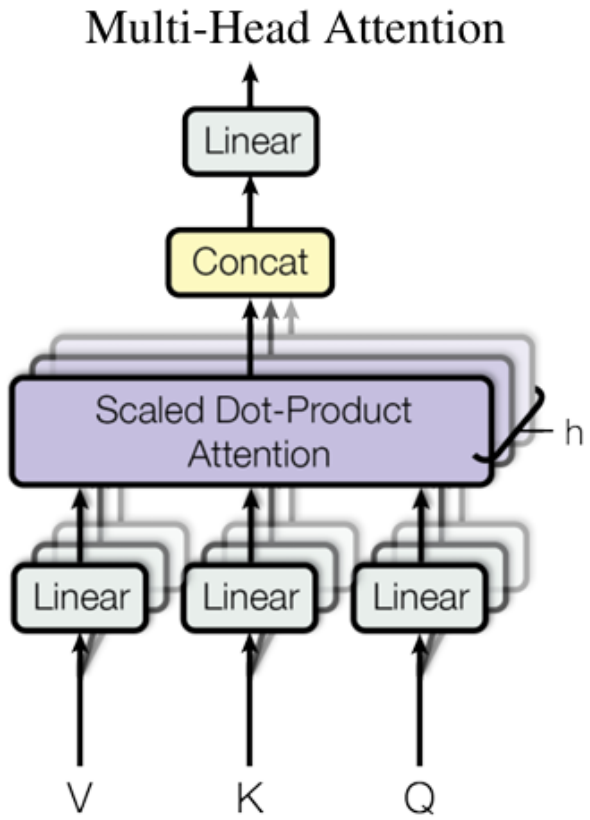
"only an im

classical apprc
representatior

Position Two

distributed re
opportunity tc
knowledge (Cl





Let's Build a Next Token Generator given a Sequence

Exploring **attention**

Sequence	Next token
a b a	b
a c b	a
a d c	c

Let's stay simple

One-hot representation of the vocabulary

a	b	c	d
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

One-hot representation of positions

1	2	3	4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

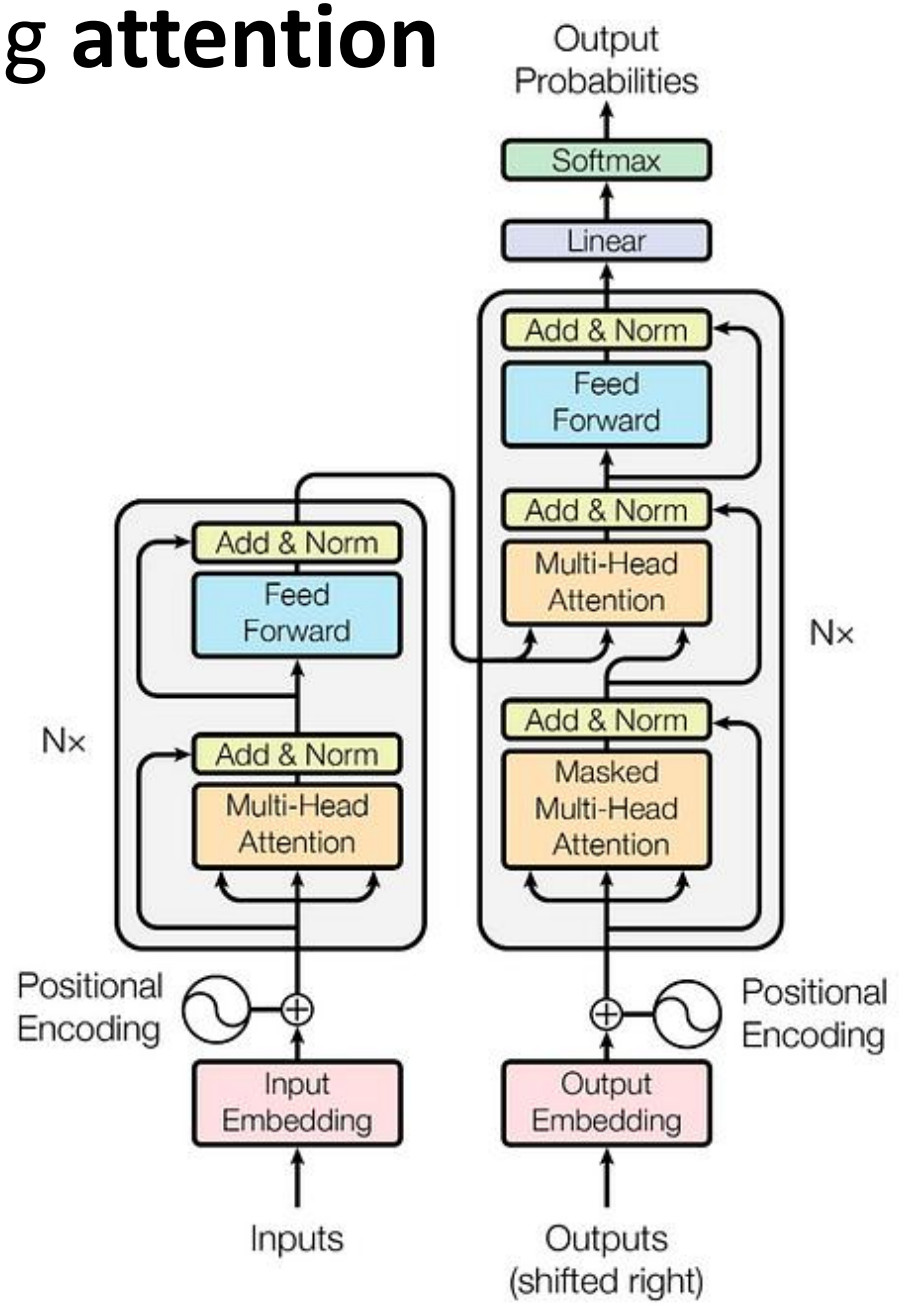


Figure 1: The Transformer - model architecture.

Let's Build a Next Token Generator given a Sequence

Sequence	Next token
a b a	b
a c b	a
a d c	c

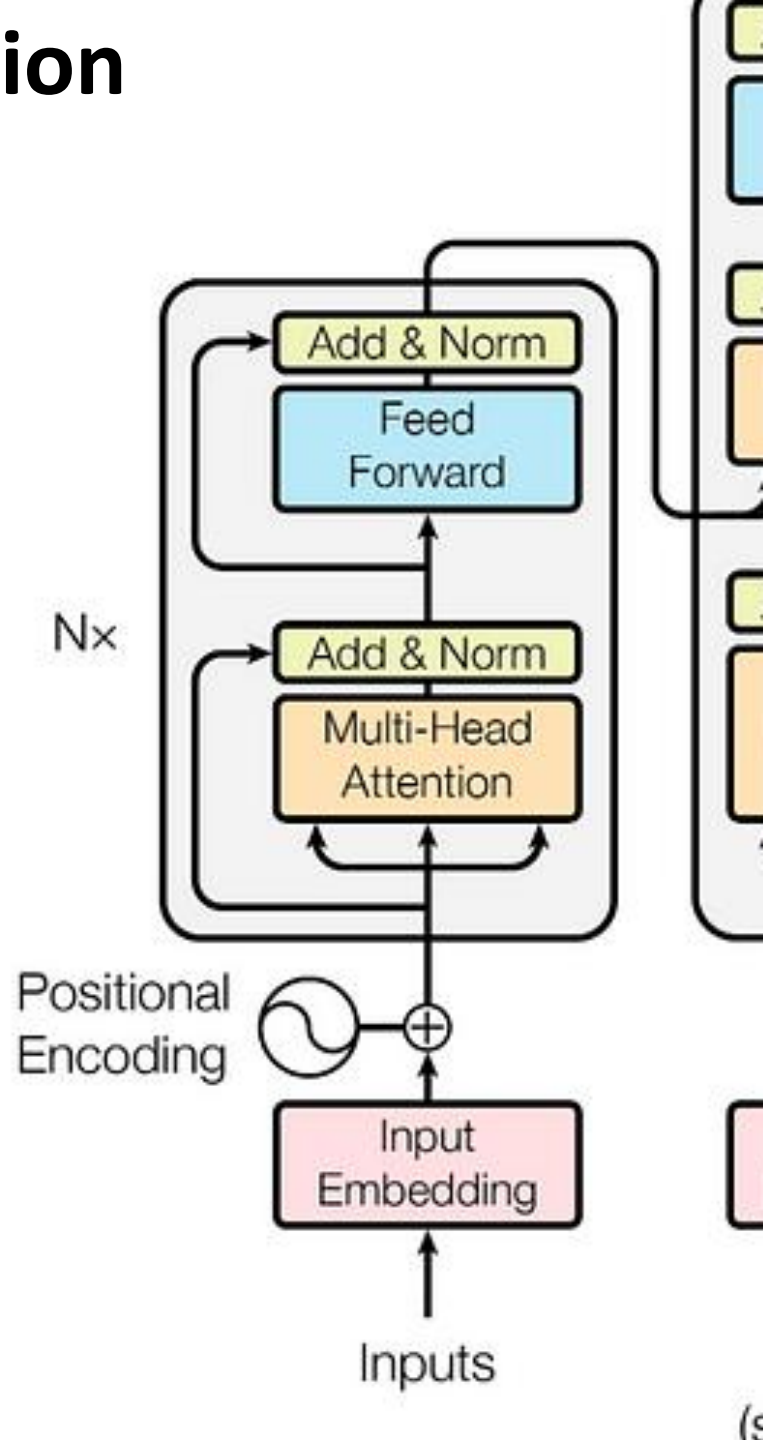
One-hot representation of the vocabulary

a	b	c	d
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

One-hot representation of positions

1	2	3	4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Exploring **attention**



Exploring **attention**

Sequence	Next token
a b a	b
a c b	a
a d c	c

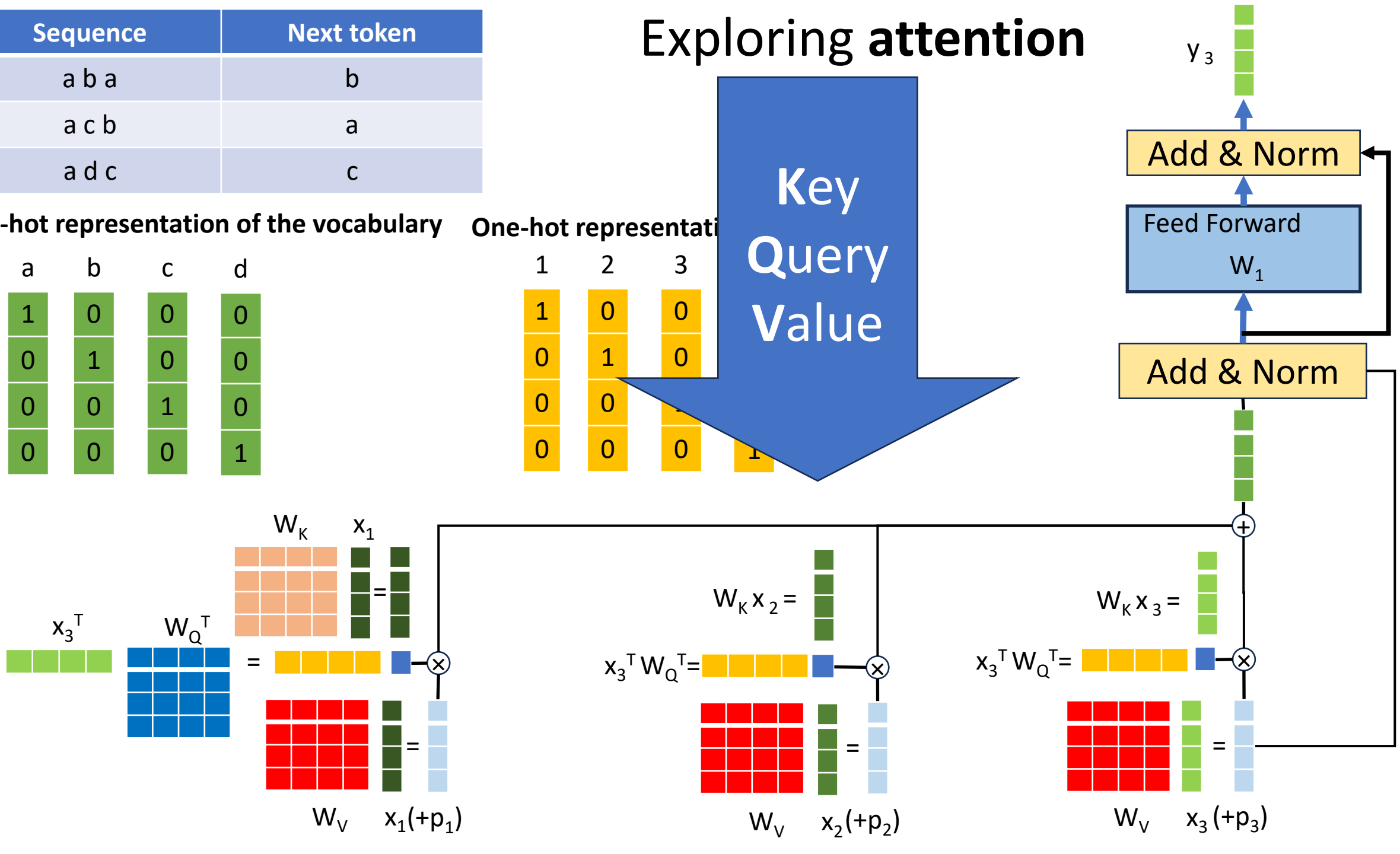
One-hot representation of the vocabulary

a	b	c	d
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

One-hot representati

1	2	3
1	0	0
0	1	0
0	0	
0	0	0

Key
Query
Value



Exploring **attention**

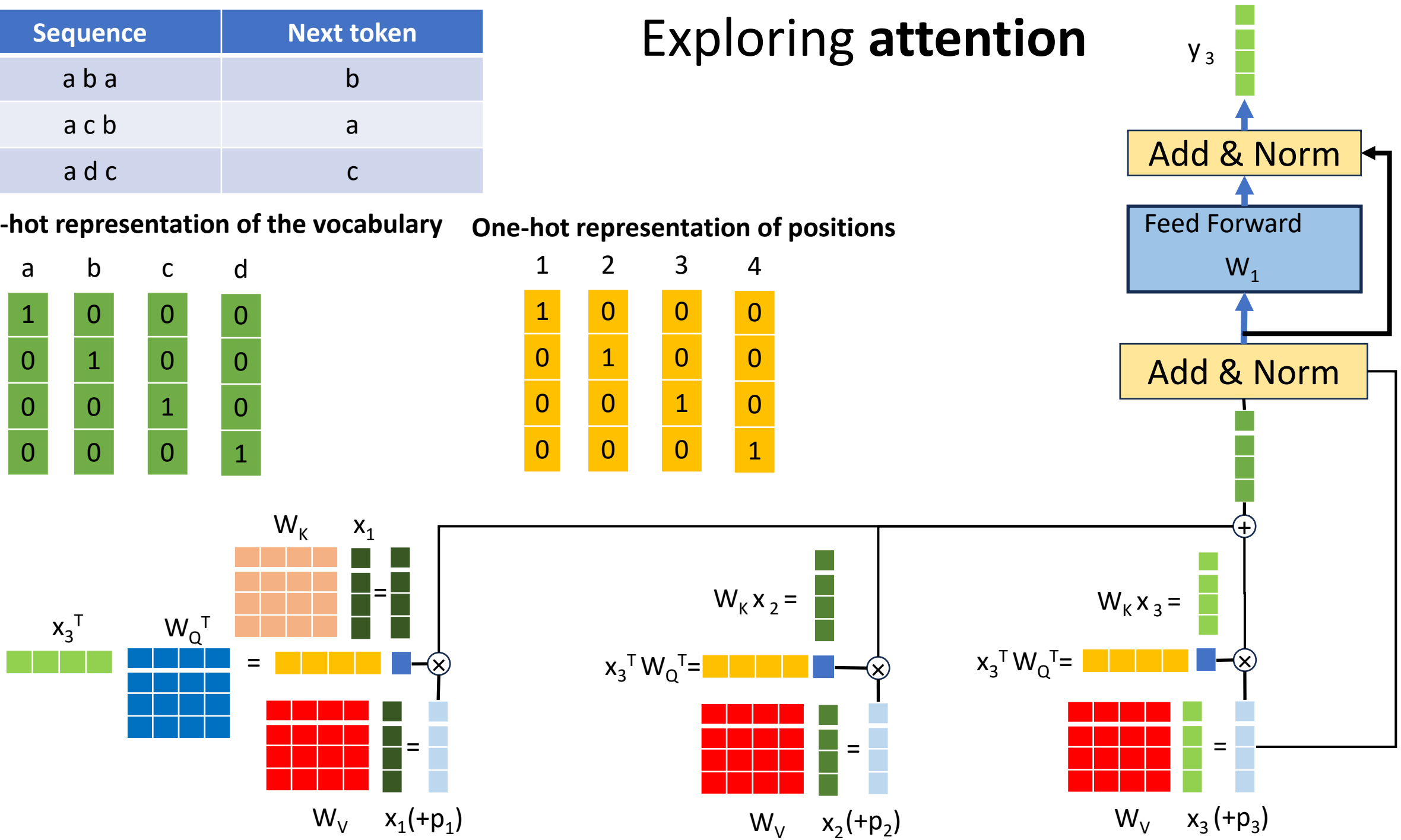
Sequence	Next token
a b a	b
a c b	a
a d c	c

One-hot representation of the vocabulary

a	b	c	d
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

One-hot representation of positions

1	2	3	4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1



Exploring **attention**

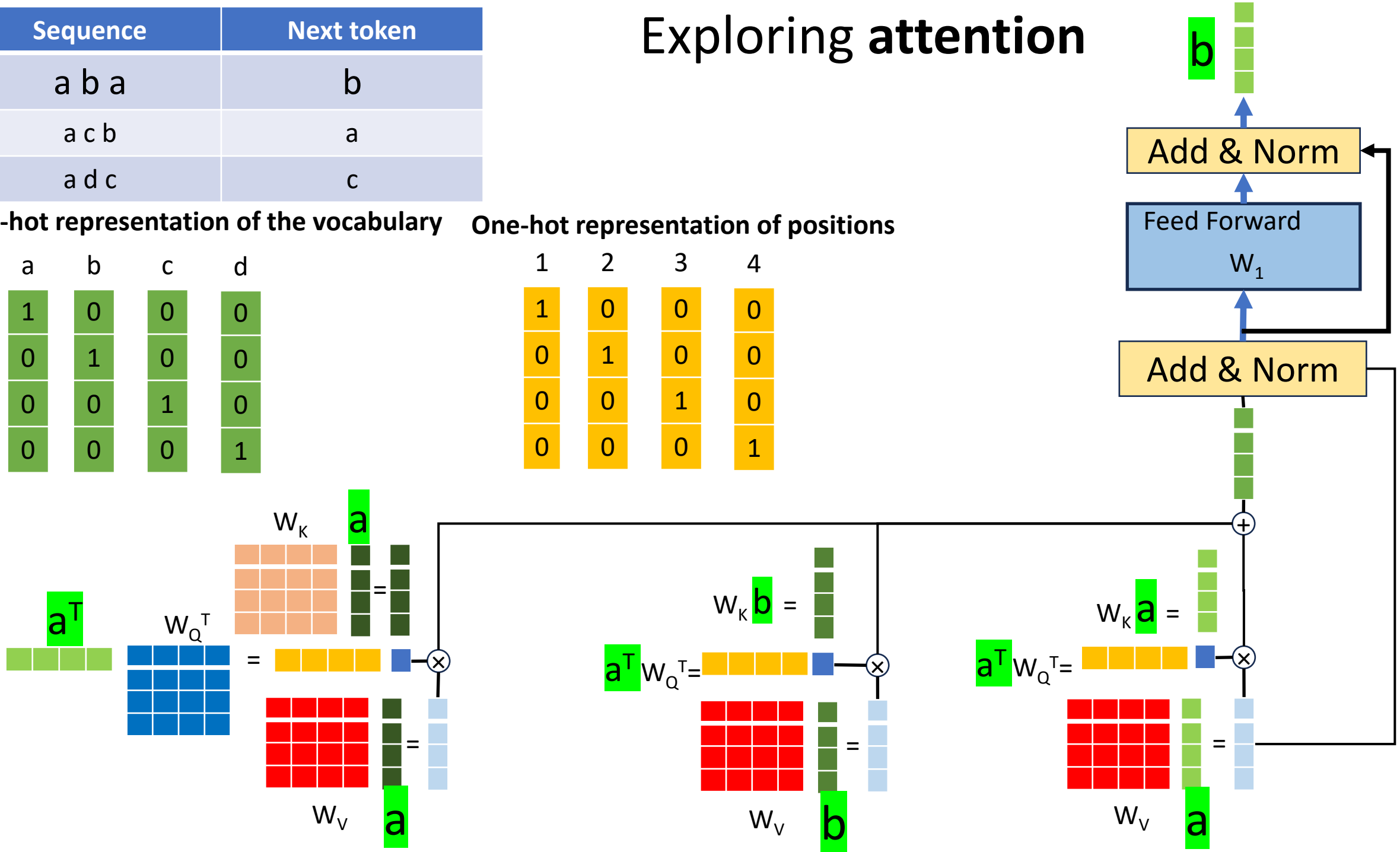
Sequence	Next token
a b a	b
a c b	a
a d c	c

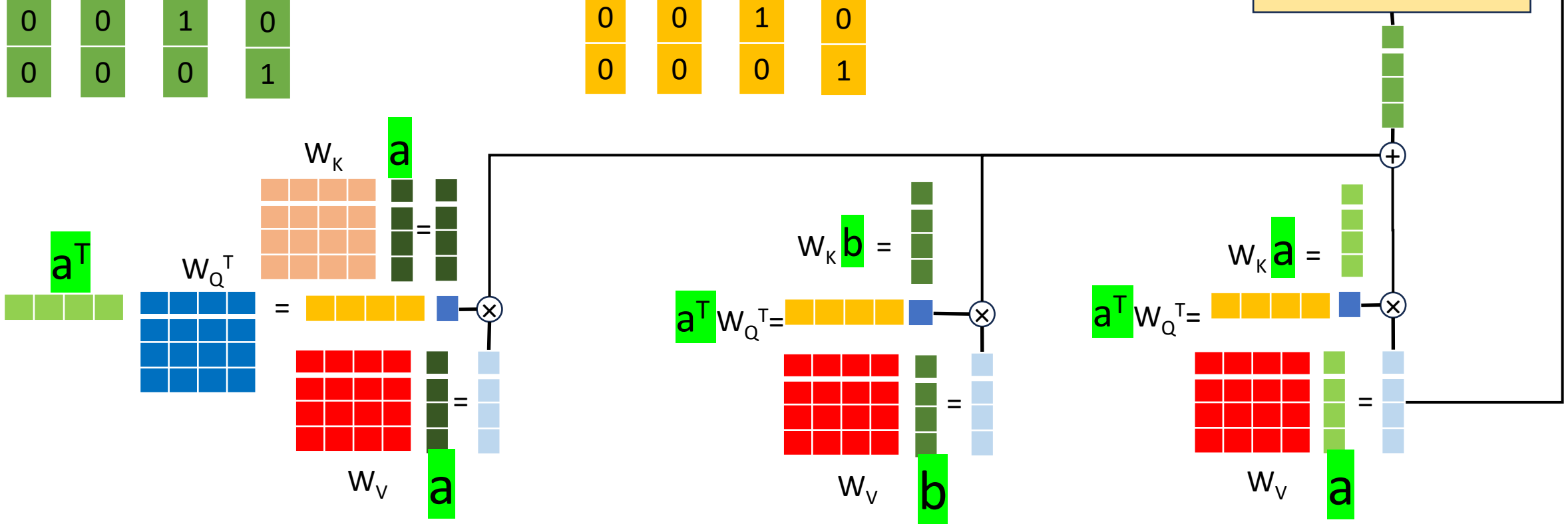
One-hot representation of the vocabulary

a	b	c	d
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

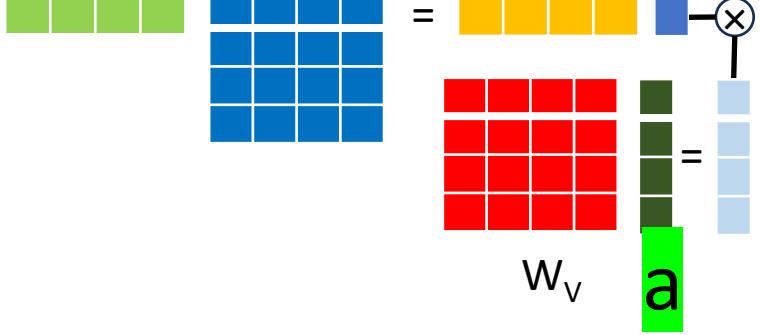
One-hot representation of positions

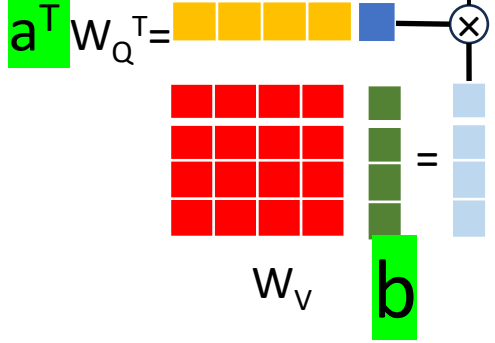
1	2	3	4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

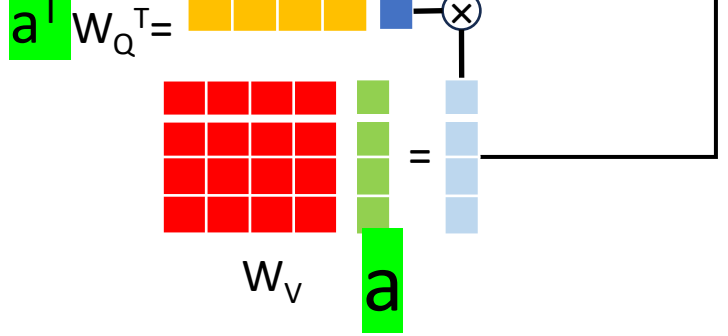




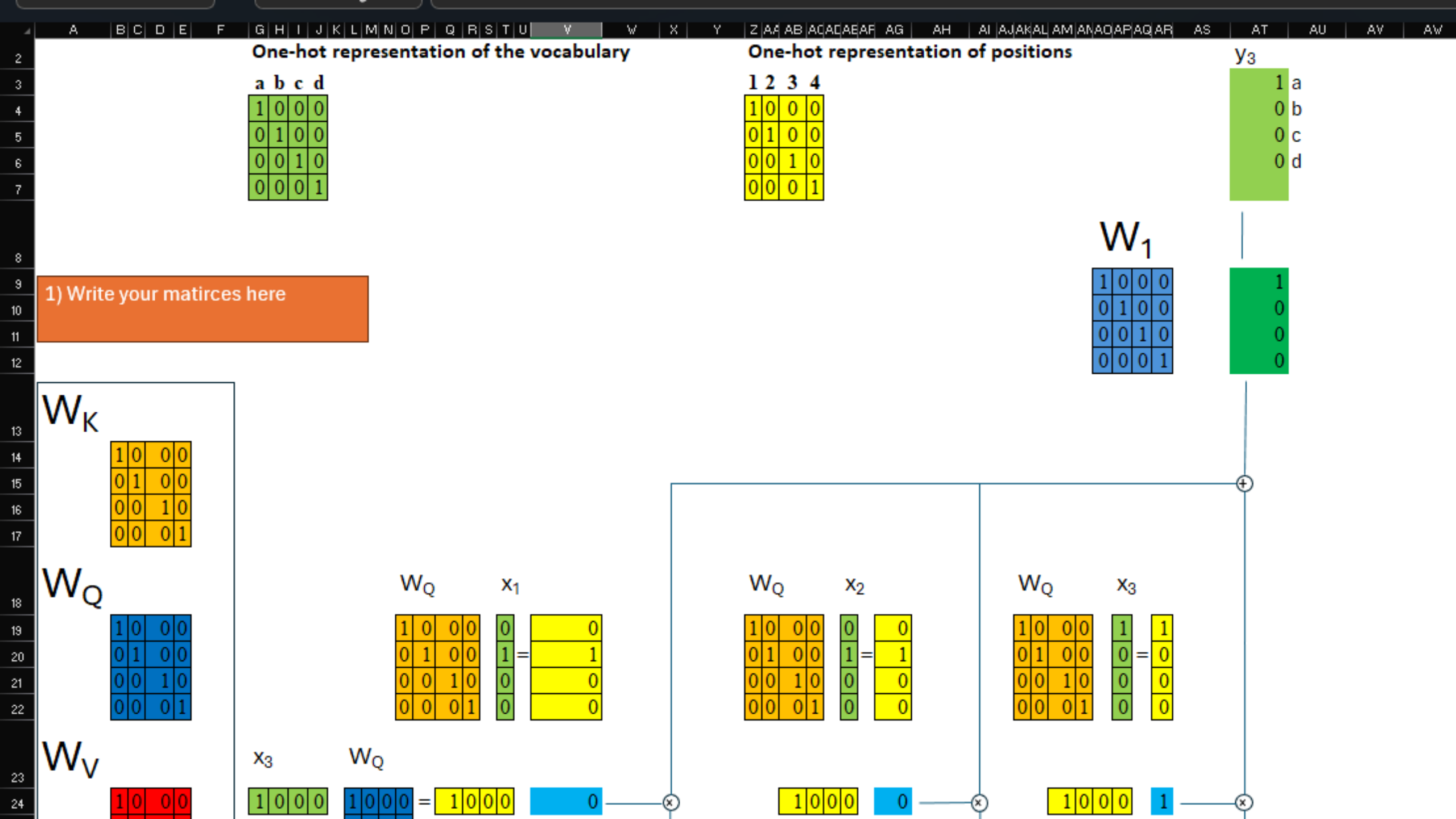
https://urly.it/31grrh



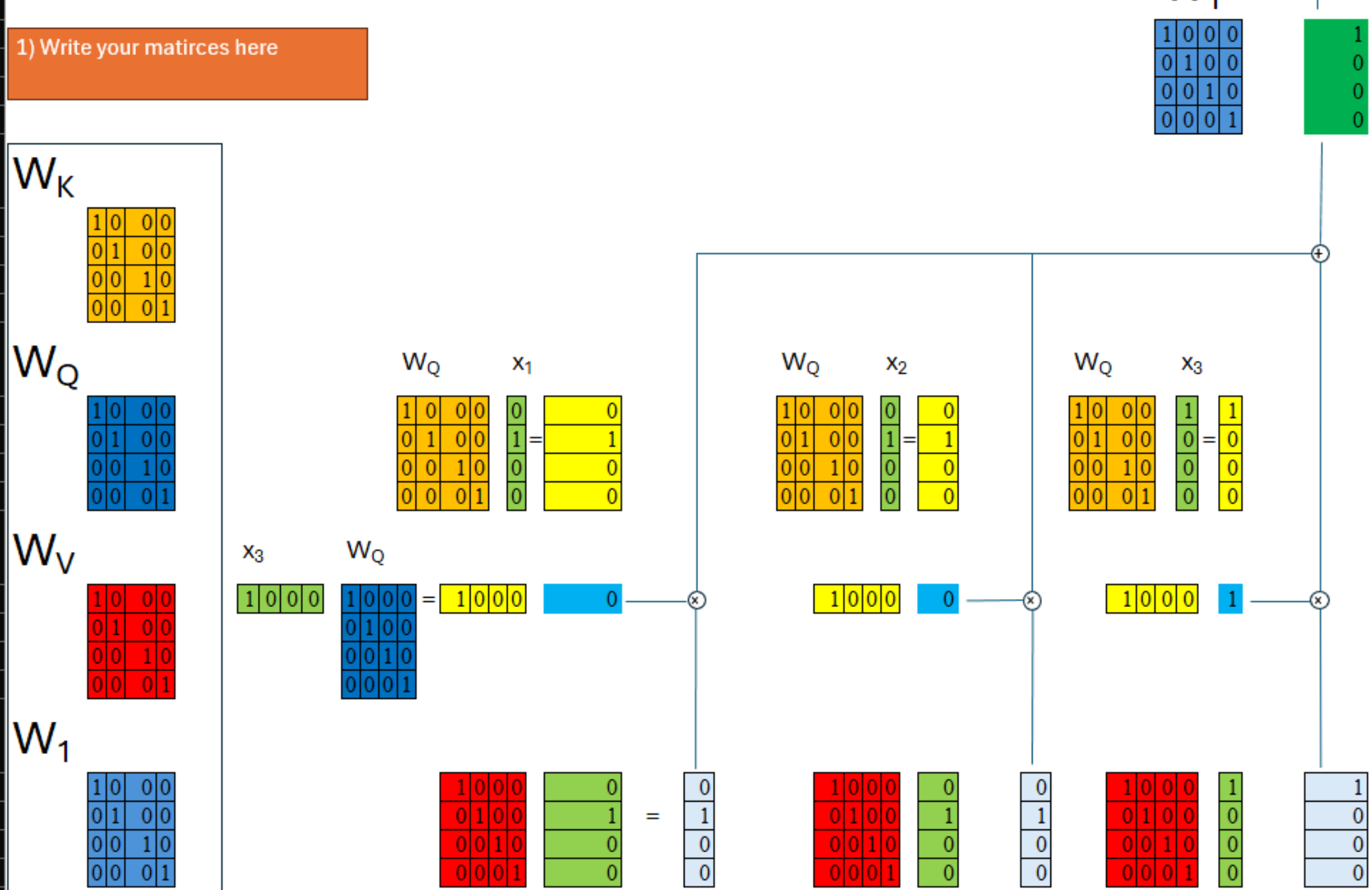


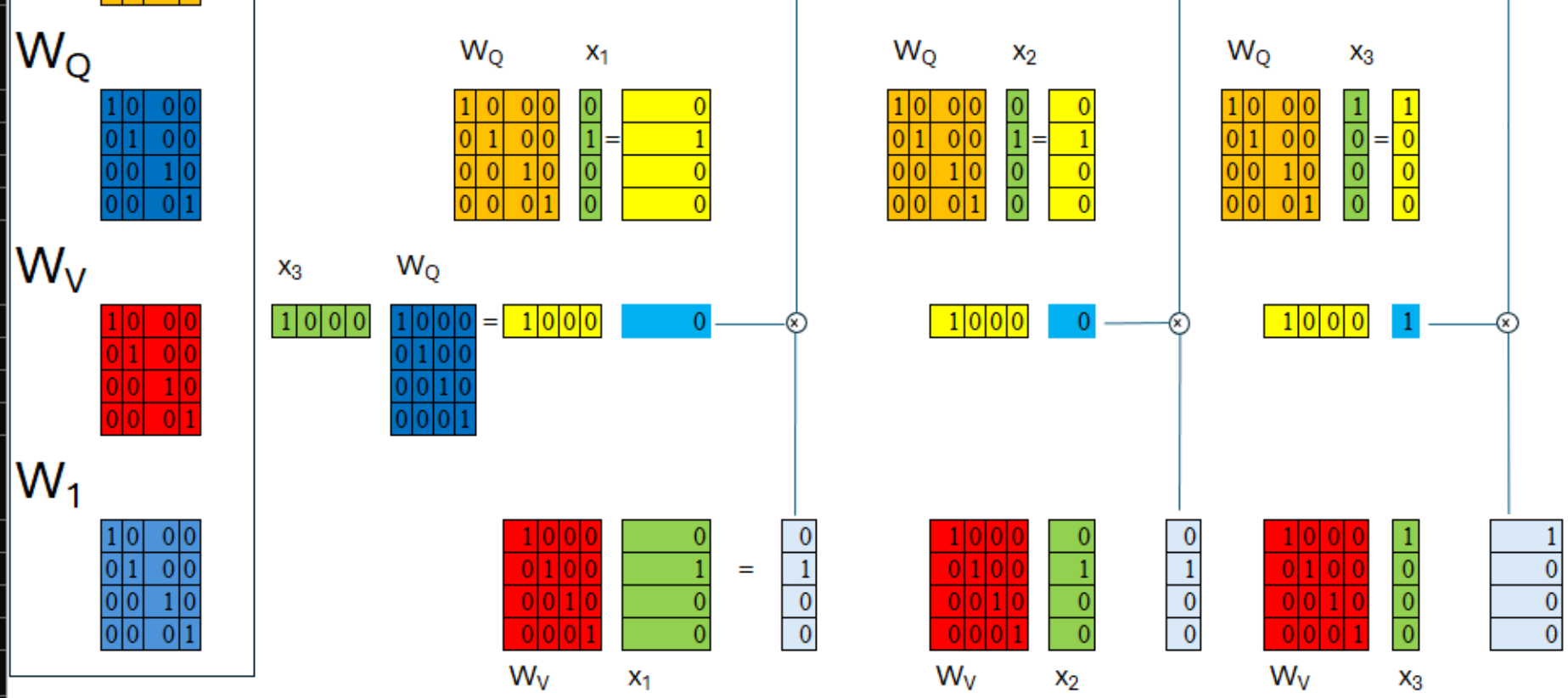


https://urly.it/31grrh


One-hot representation of the vocabulary
a b c d
One-hot representation of positions
1 2 3 4
y3
1 a
0 b
0 c
0 d
W1
1) Write your matirces here
WK
WQ
WV
WQ x1
WQ x2
WQ x3
x3 WQ
=

1) Write your matirces here





2) Change the input by using the table input, expected output

INPUT: b b a

3) Check the output

OUTPUT: a

Input (SEQUENCE			*Expected output*
a	b	a	b
a	a	b	a
b	a	a	c

Exploring **attention**

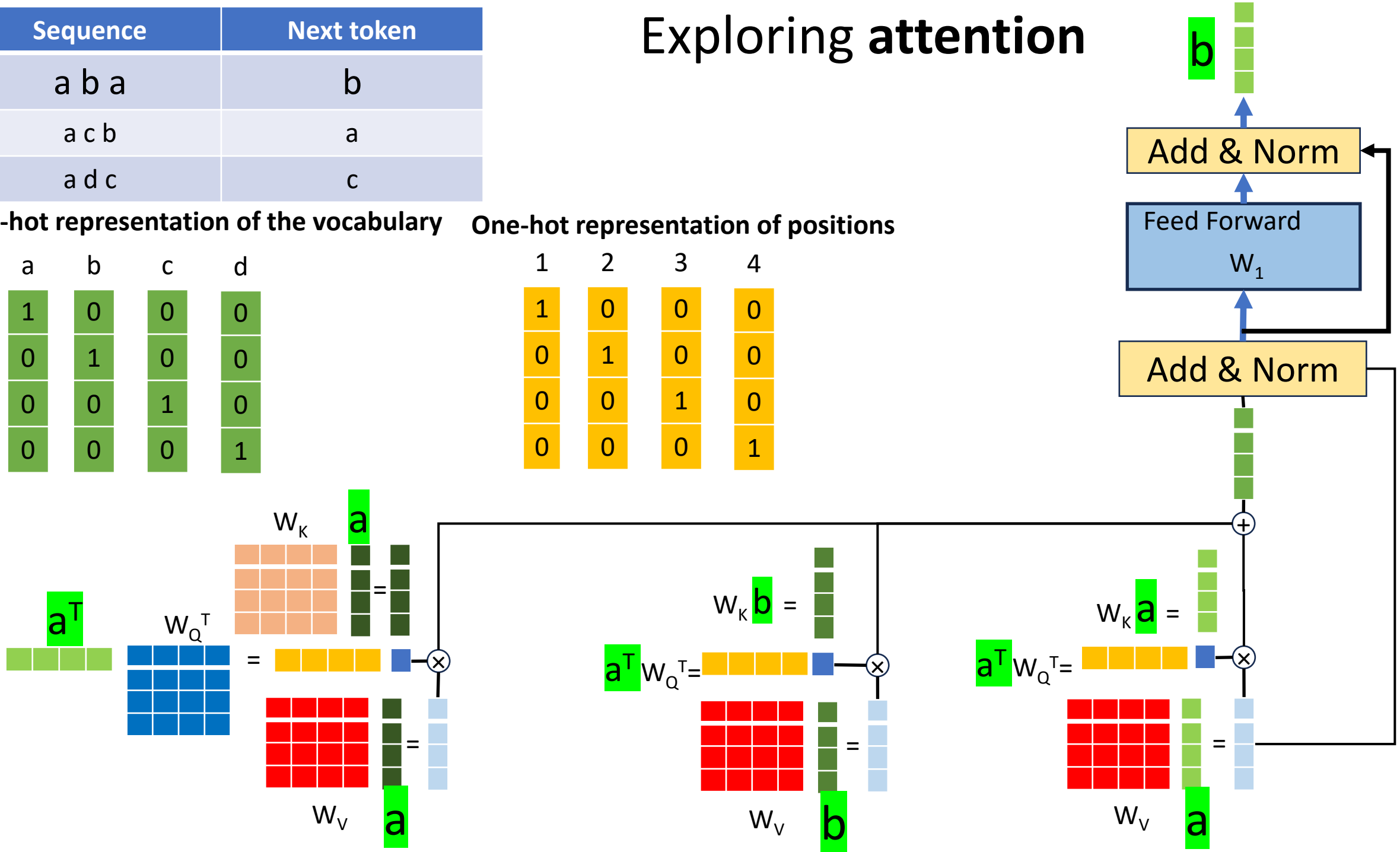
Sequence	Next token
a b a	b
a c b	a
a d c	c

One-hot representation of the vocabulary

a	b	c	d
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

One-hot representation of positions

1	2	3	4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

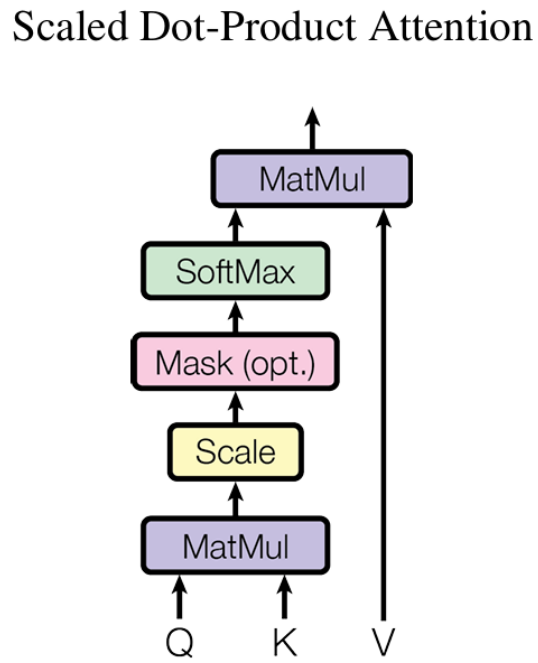


Exploring **attention**

Let's Build a Next Token Generator given a Sequence

Sequence	Next token
a b a	b
a c b	a
a d c	c

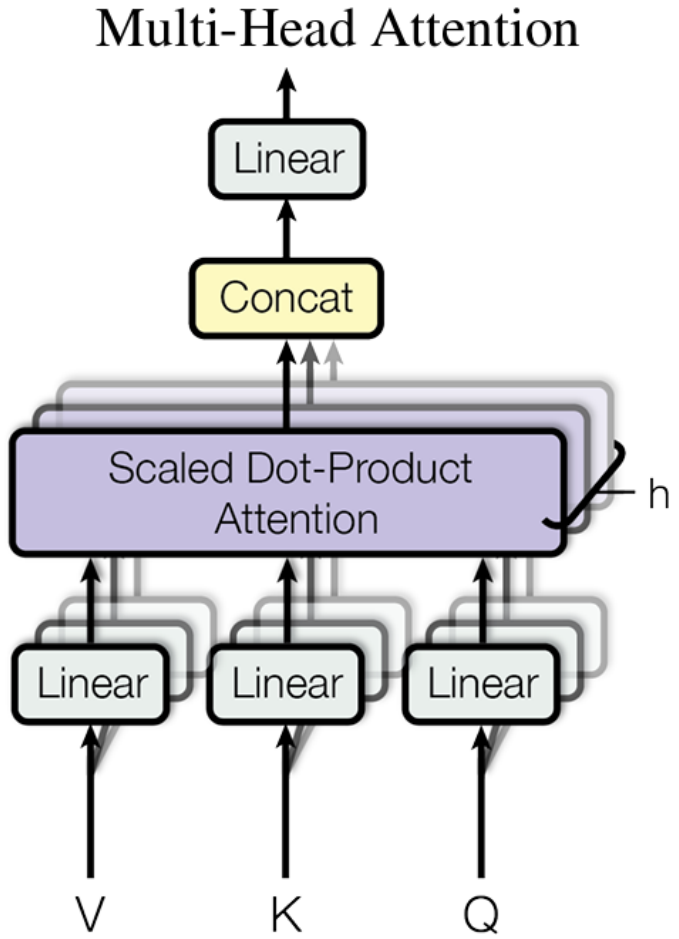
Apparently, it is not possible!



Exploring **attention**

Let's Build a Next Token Generator given a Sequence

Sequence	Next token
a b a	b
a c b	a
a d c	c



Exploring **multi-head attention**

Exploring **multi-head attention**

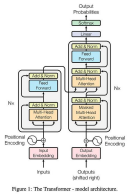
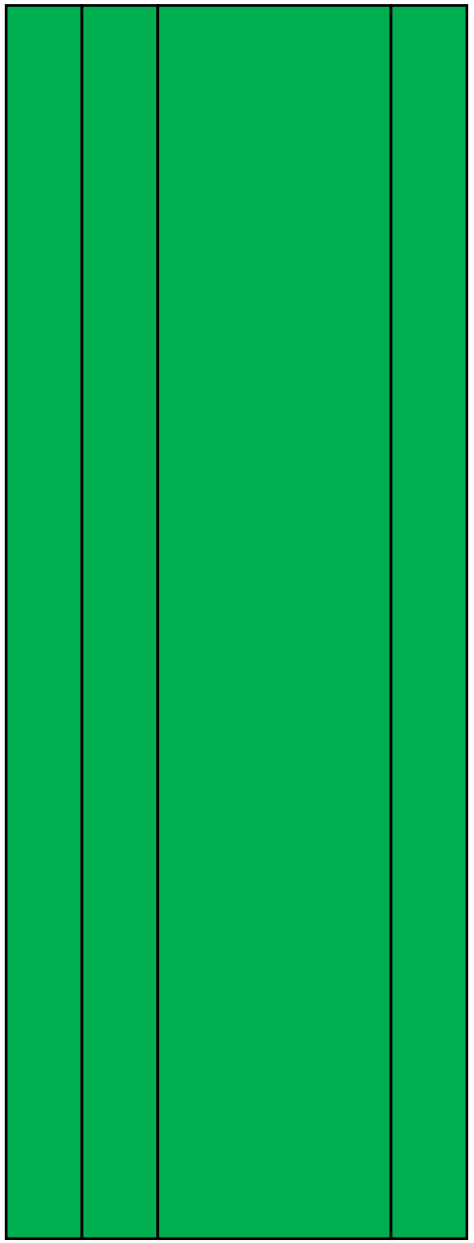


Figure 1: The Transformer - model architecture.

Embedding



Exploring **multi-head attention**

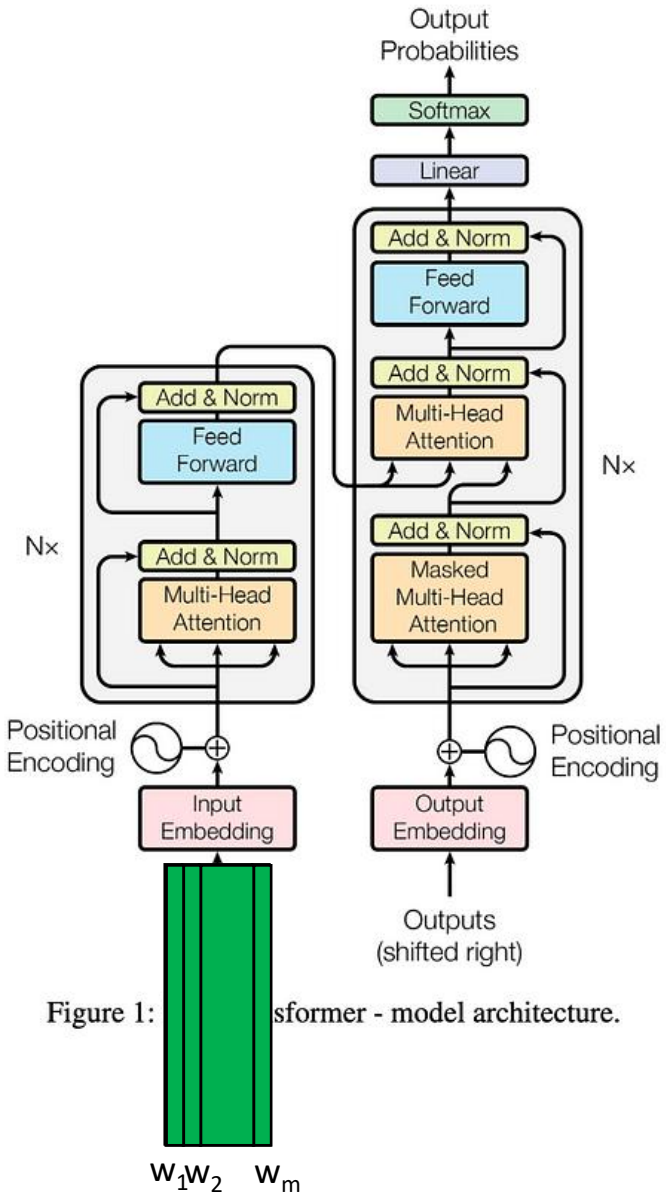
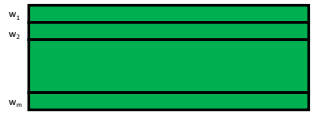


Figure 1: sformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**



m × d



m × d



m × d

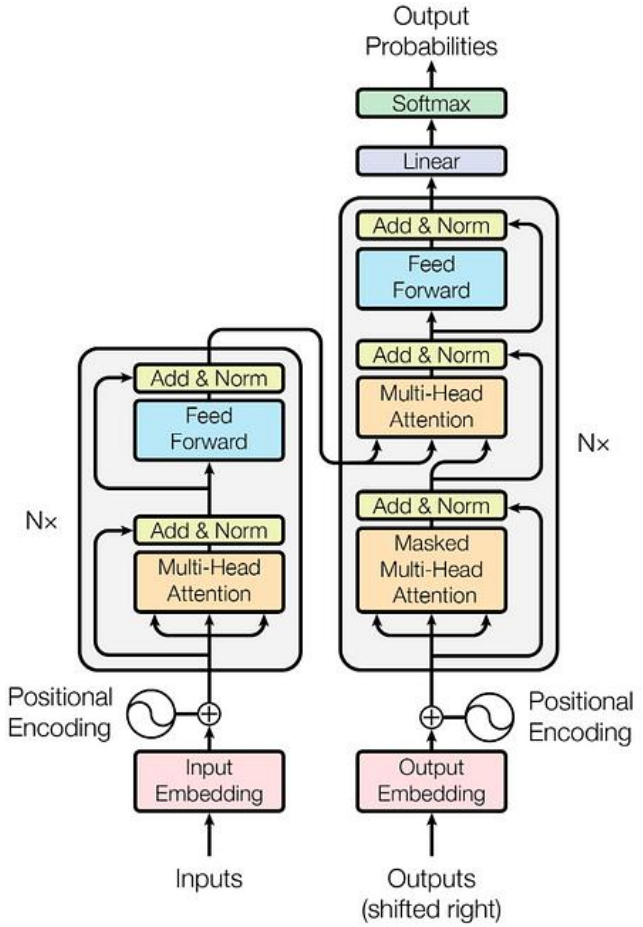
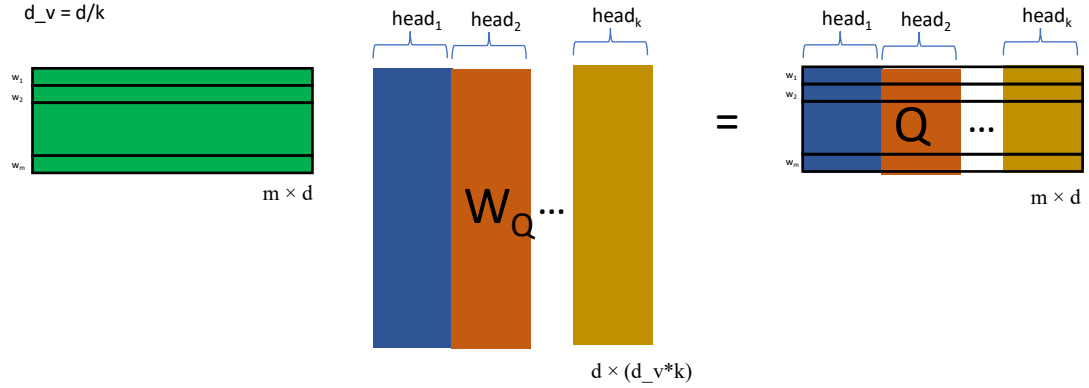
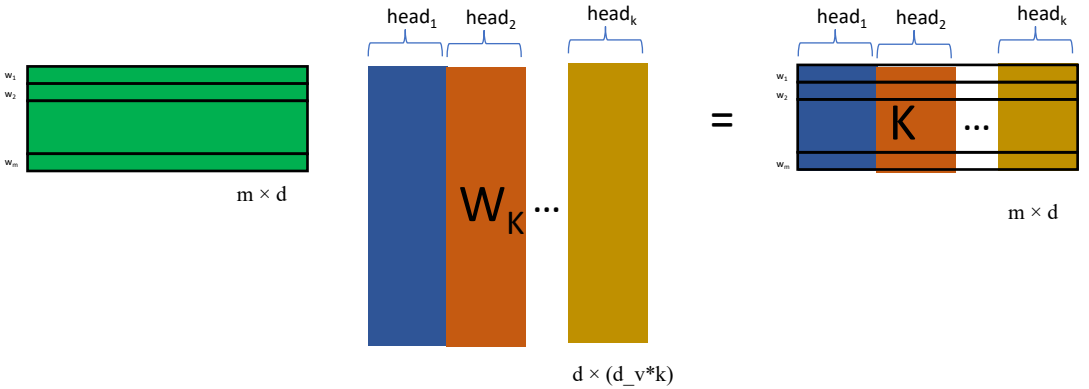


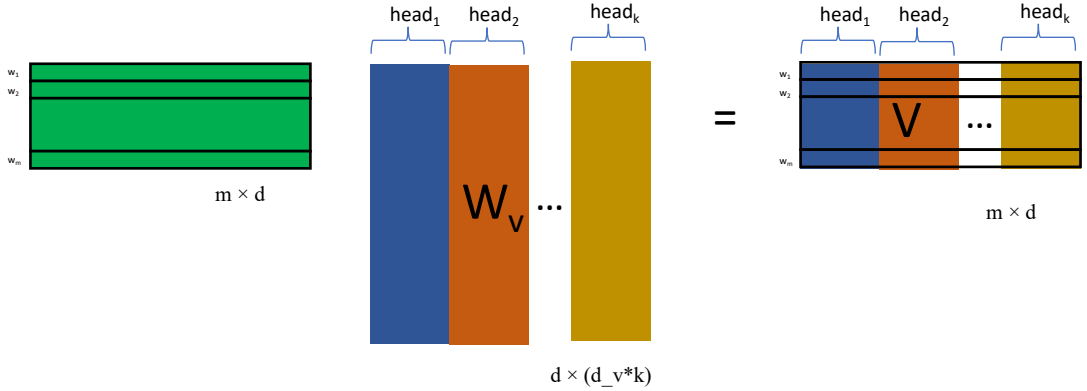
Figure 1: The Transformer - model architecture.

Exploring **multi-head attention**

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k







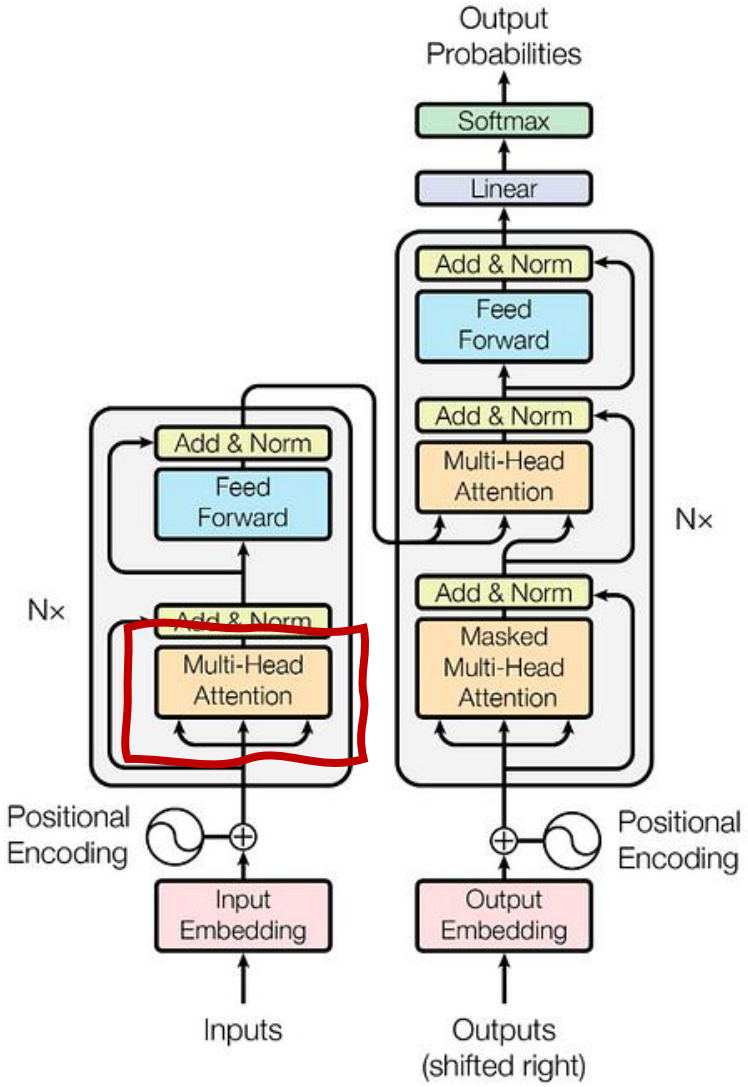
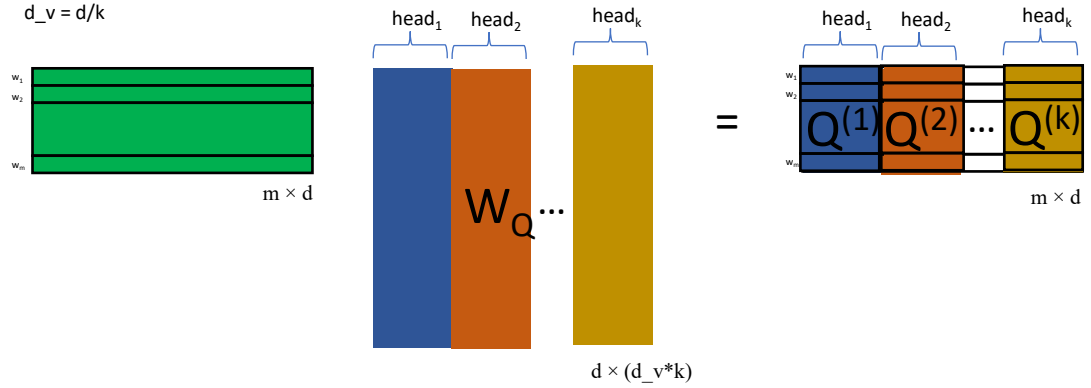
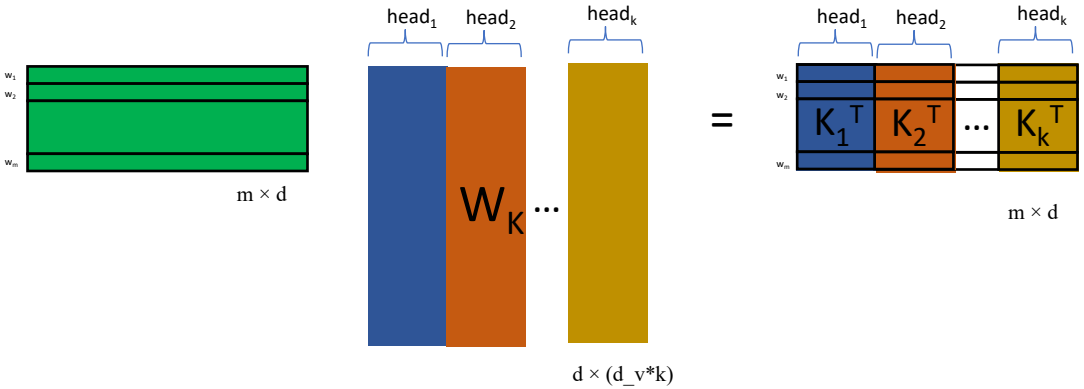


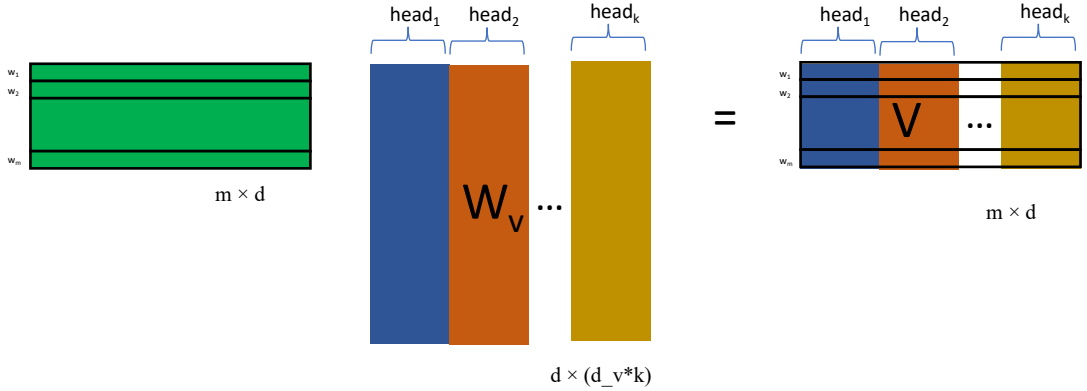
Figure 1: The Transformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**







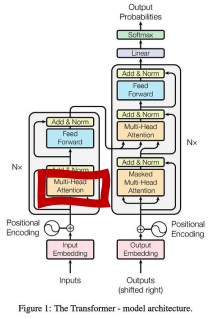
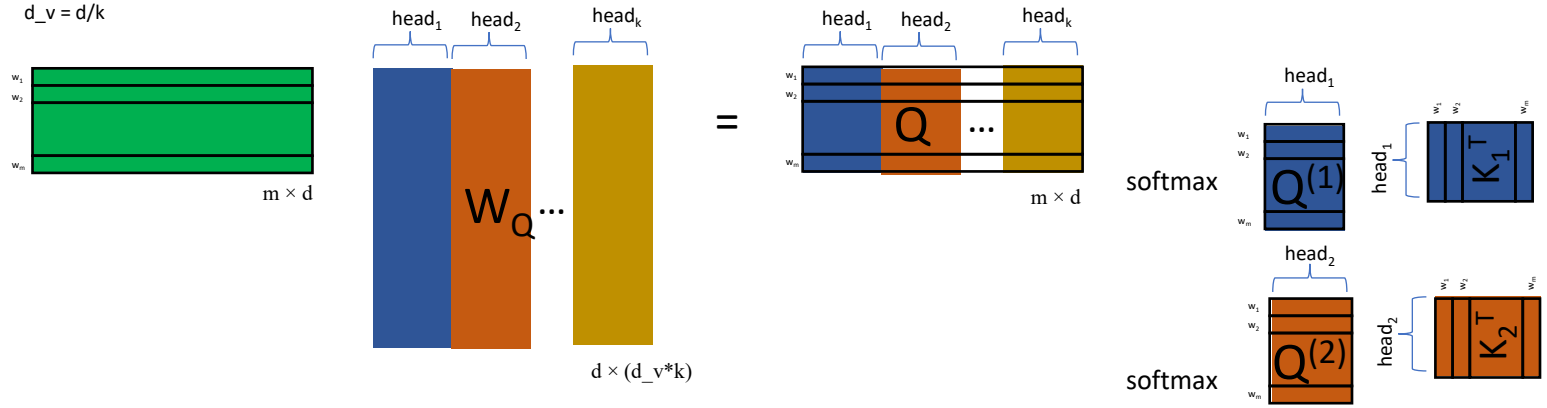


Figure 1: The Transformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**



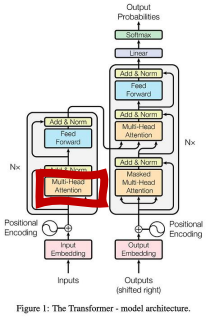
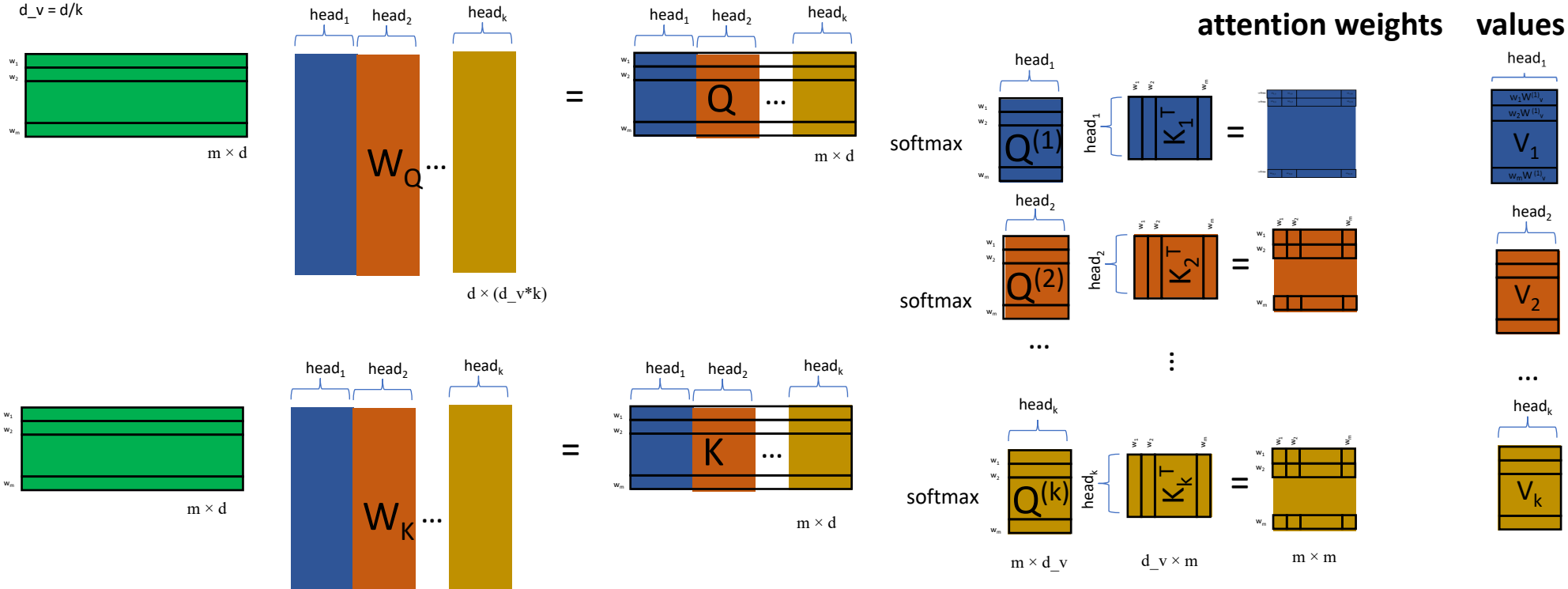


Figure 1: The Transformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**



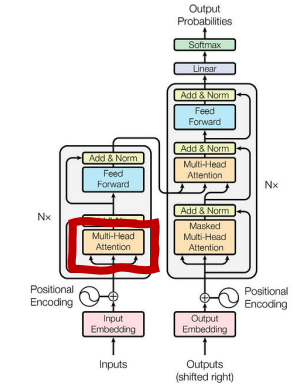
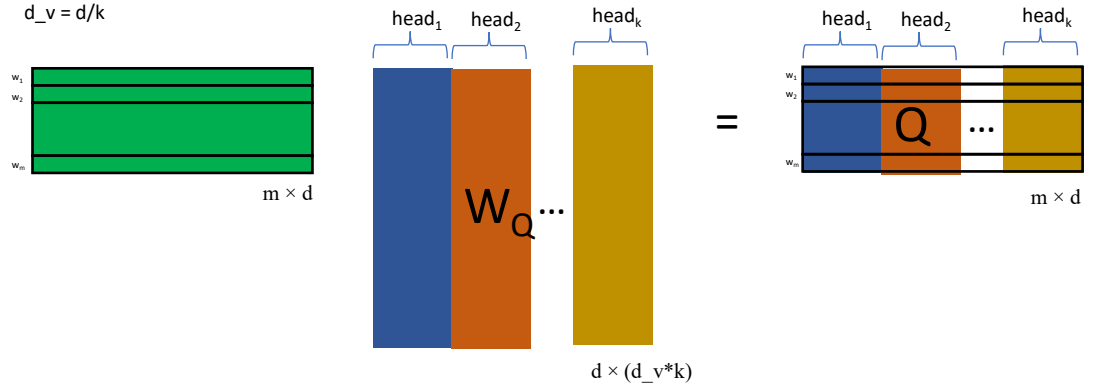
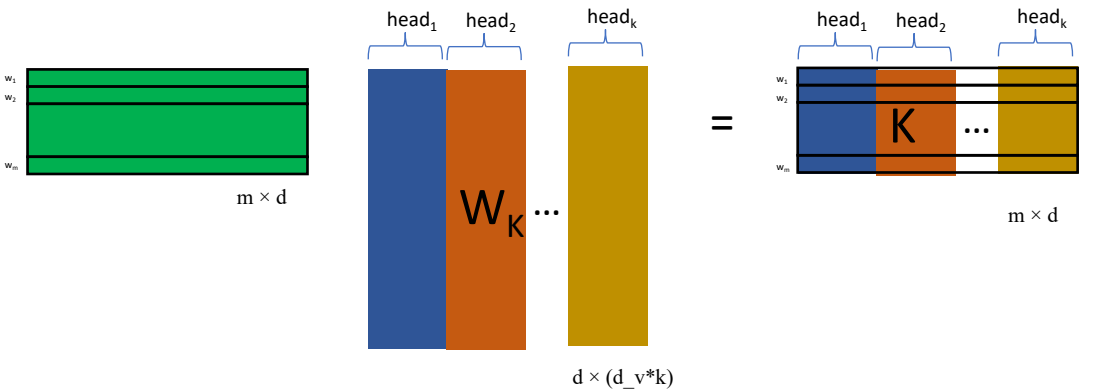


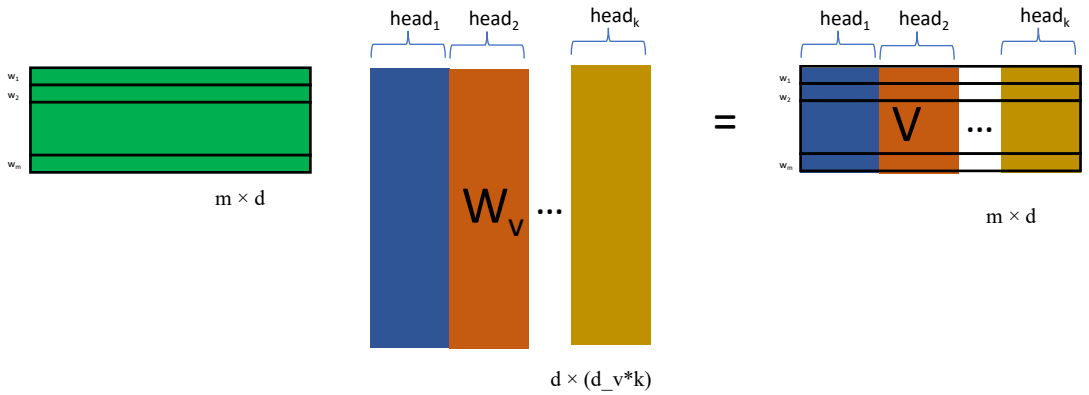
Figure 1: The Transformer - model architecture.

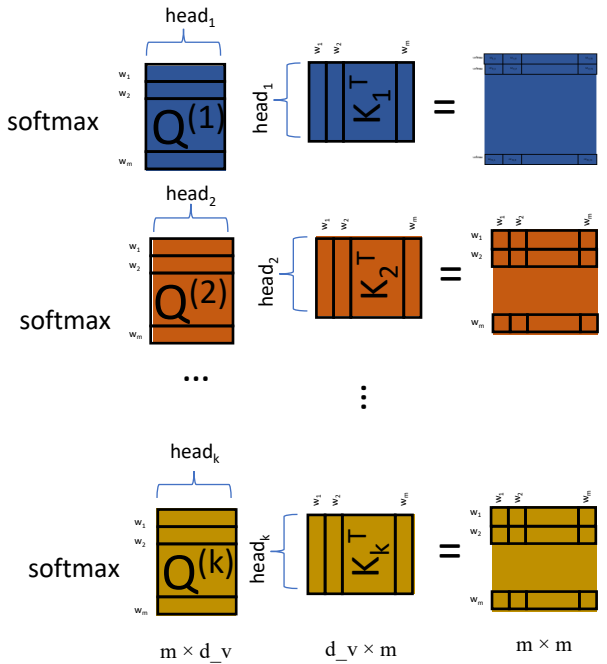
d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

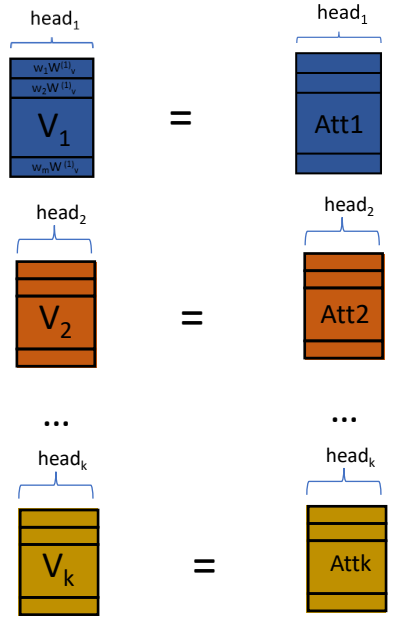
Exploring **multi-head attention**











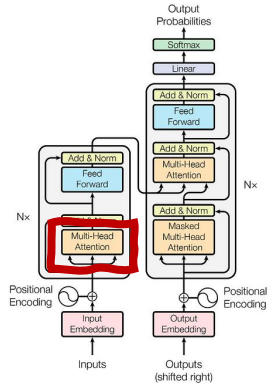
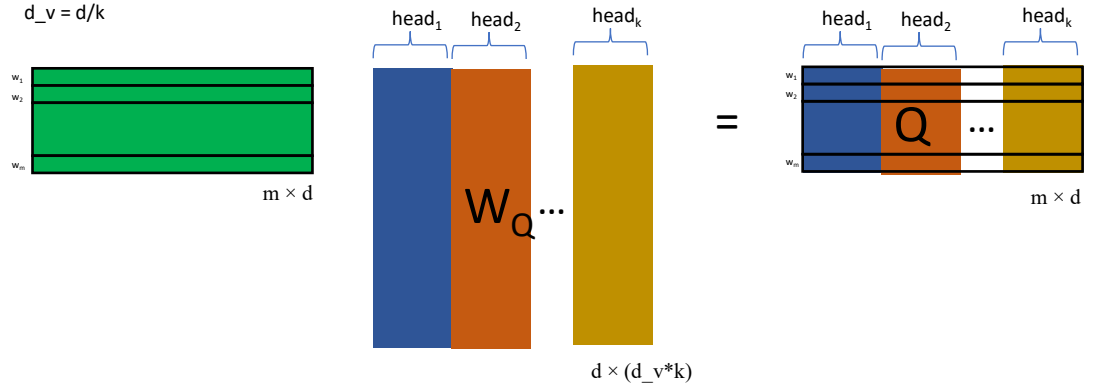
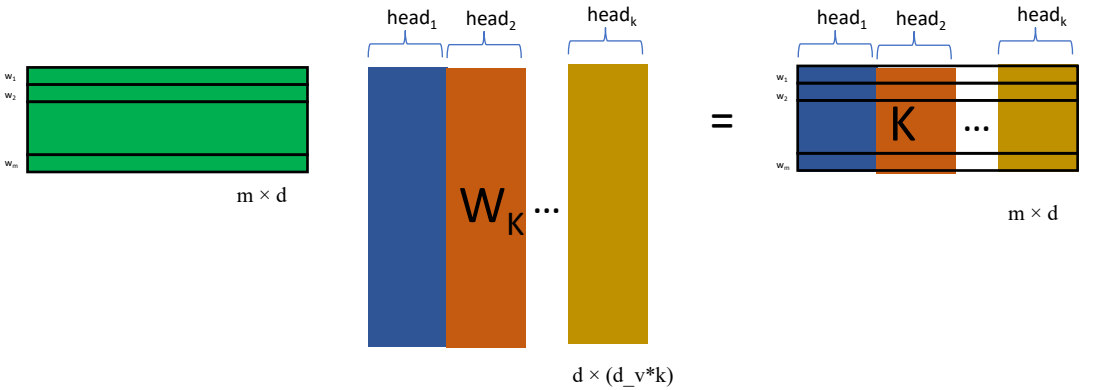


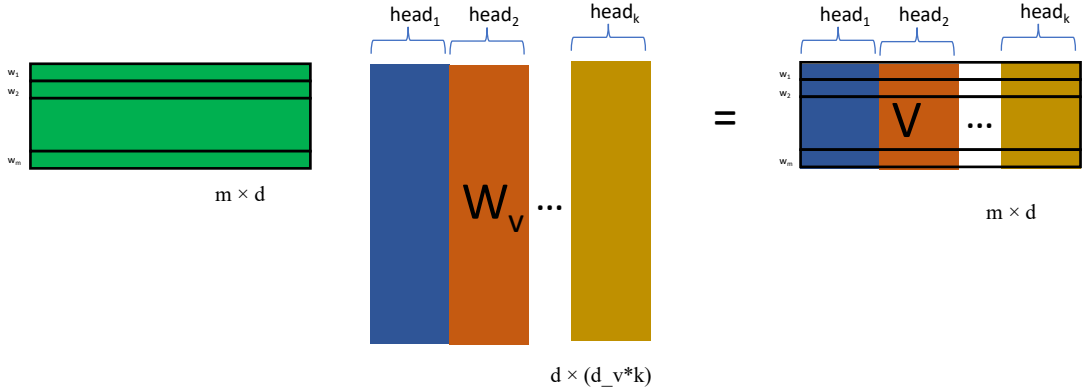
Figure 1: The Transformer - model architecture.

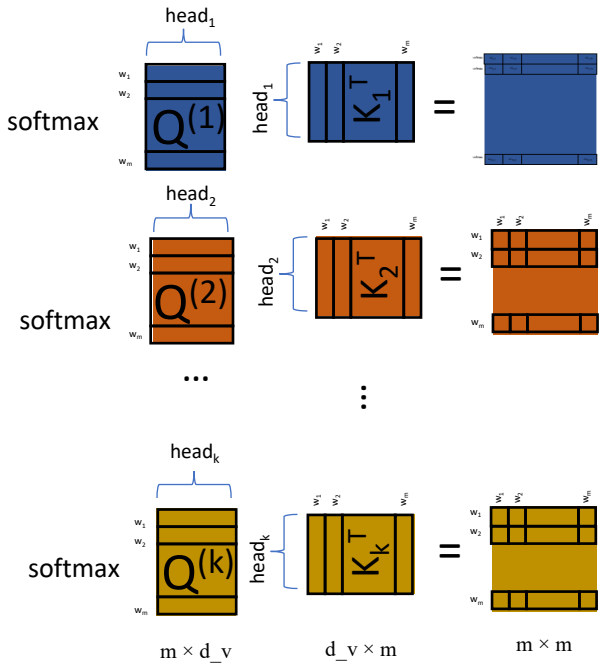
d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

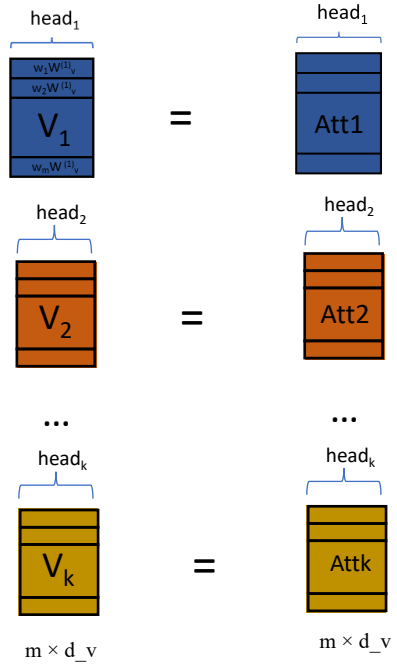
Exploring **multi-head attention**











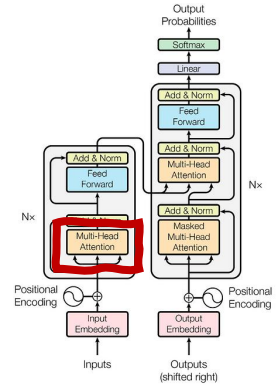
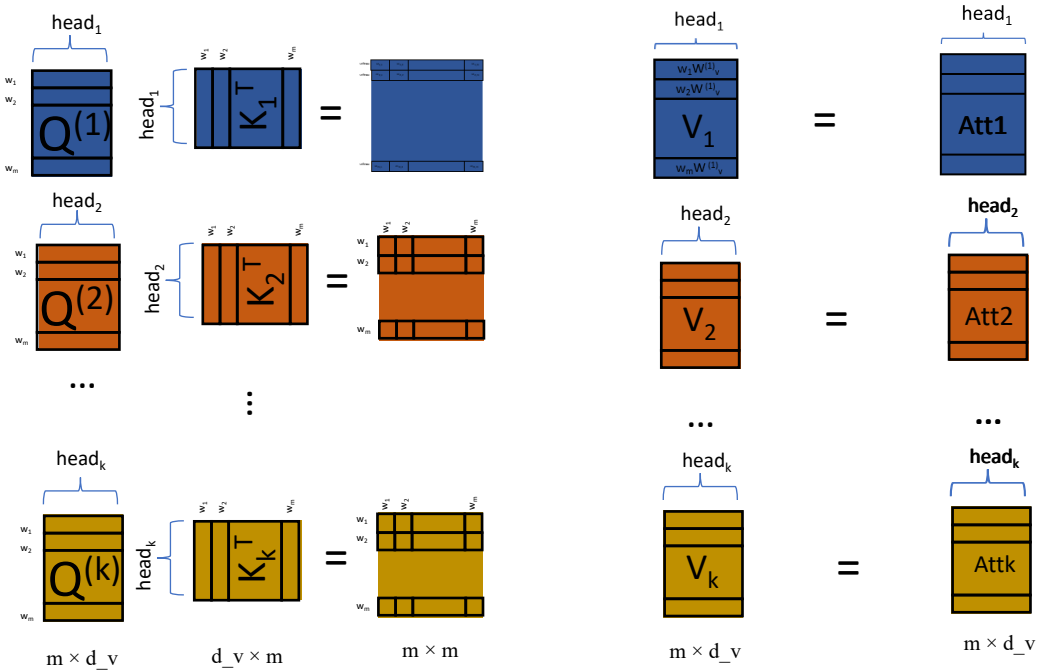


Figure 1: The Transformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**



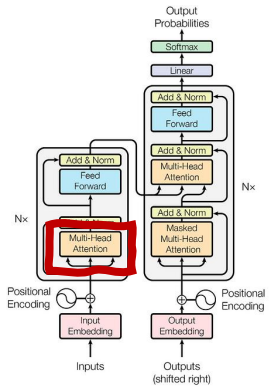
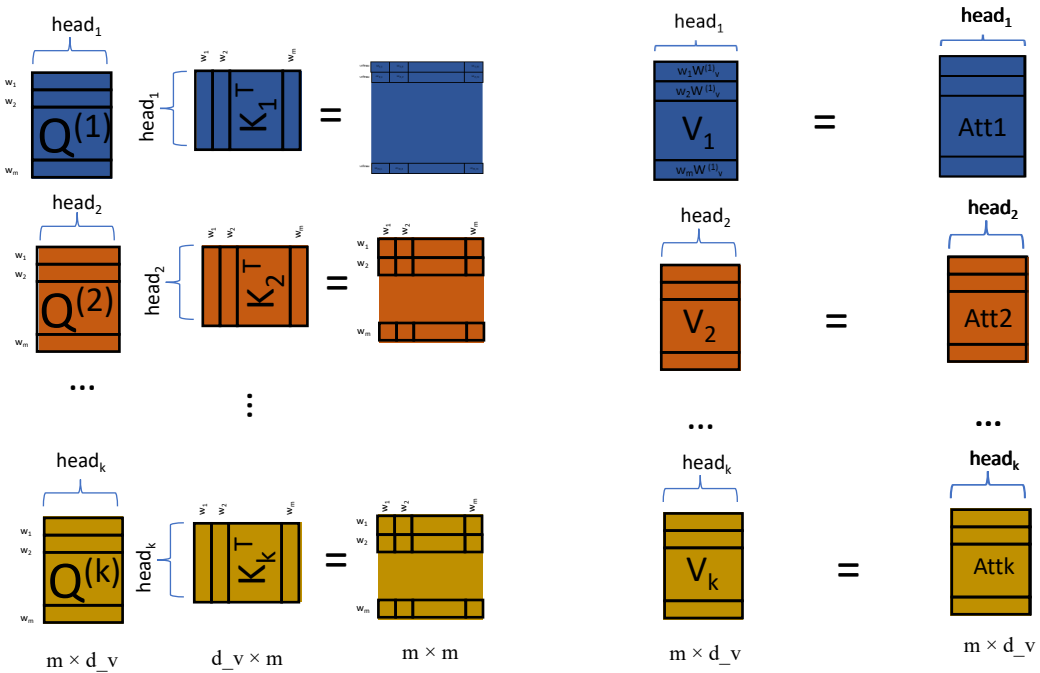


Figure 1: The Transformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**



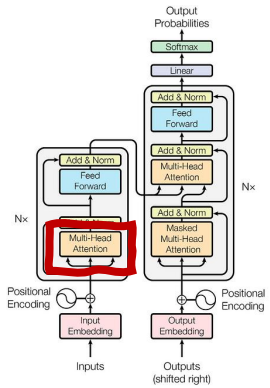
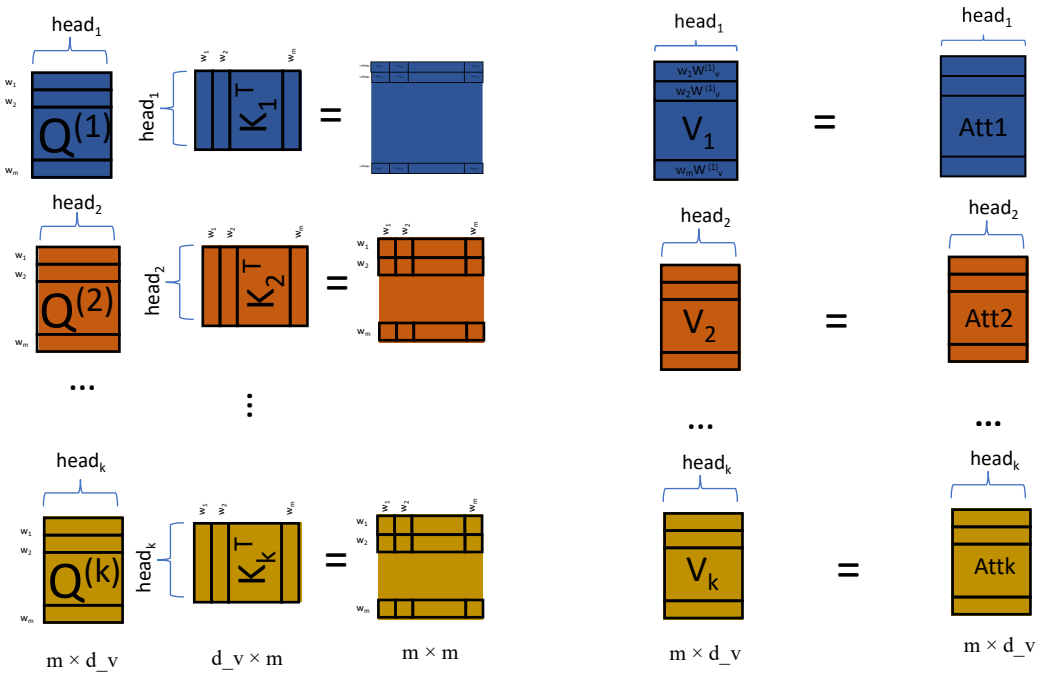
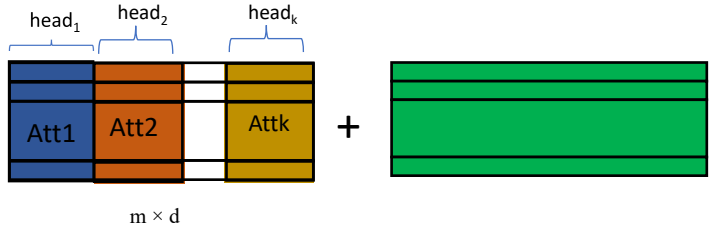


Figure 1: The Transformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**





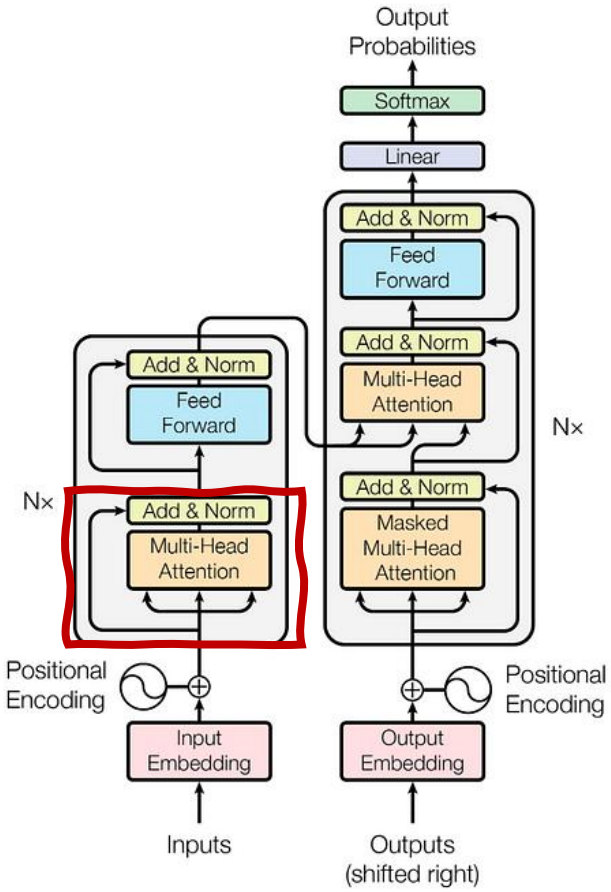


Figure 1: The Transformer - model architecture.

d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**

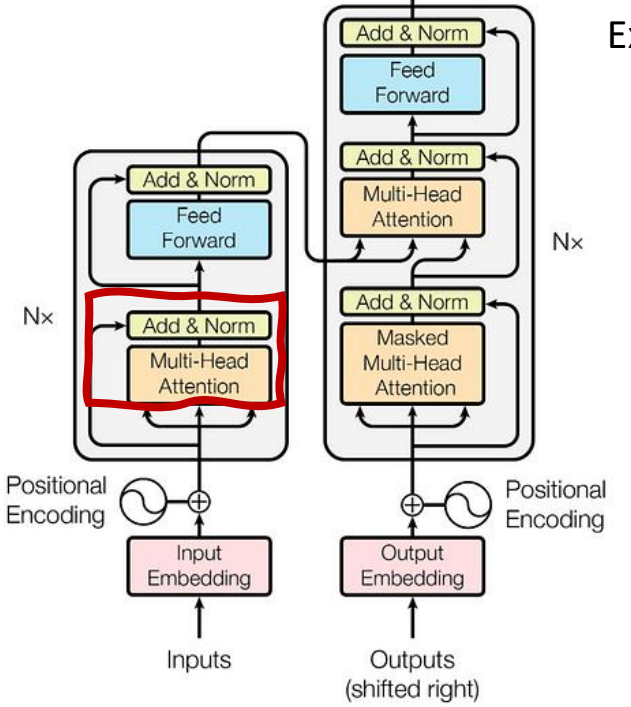
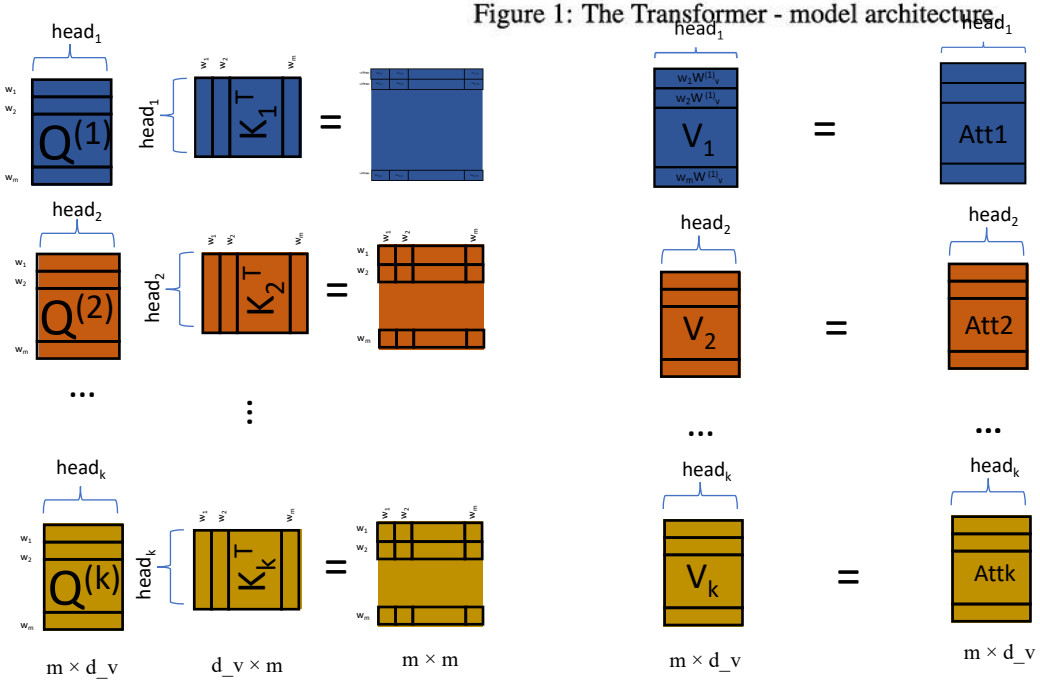
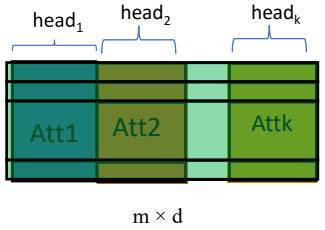


Figure 1: The Transformer - model architecture.





d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k

Exploring **multi-head attention**

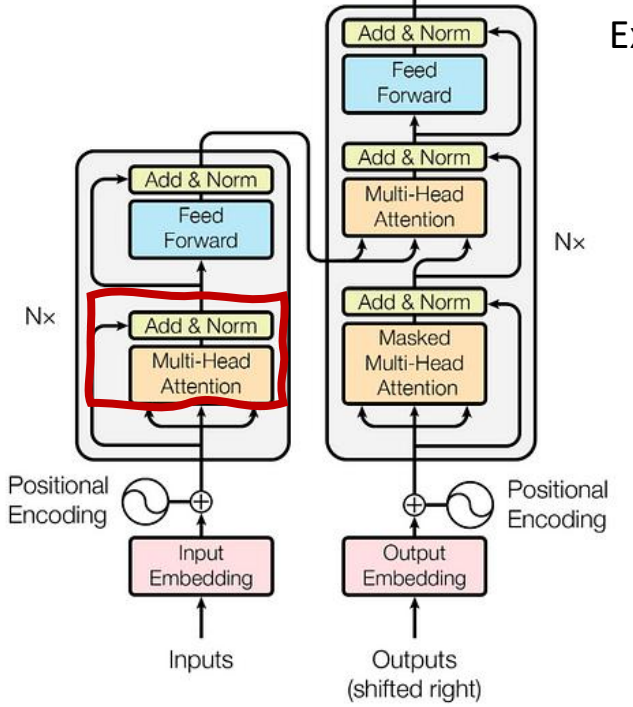
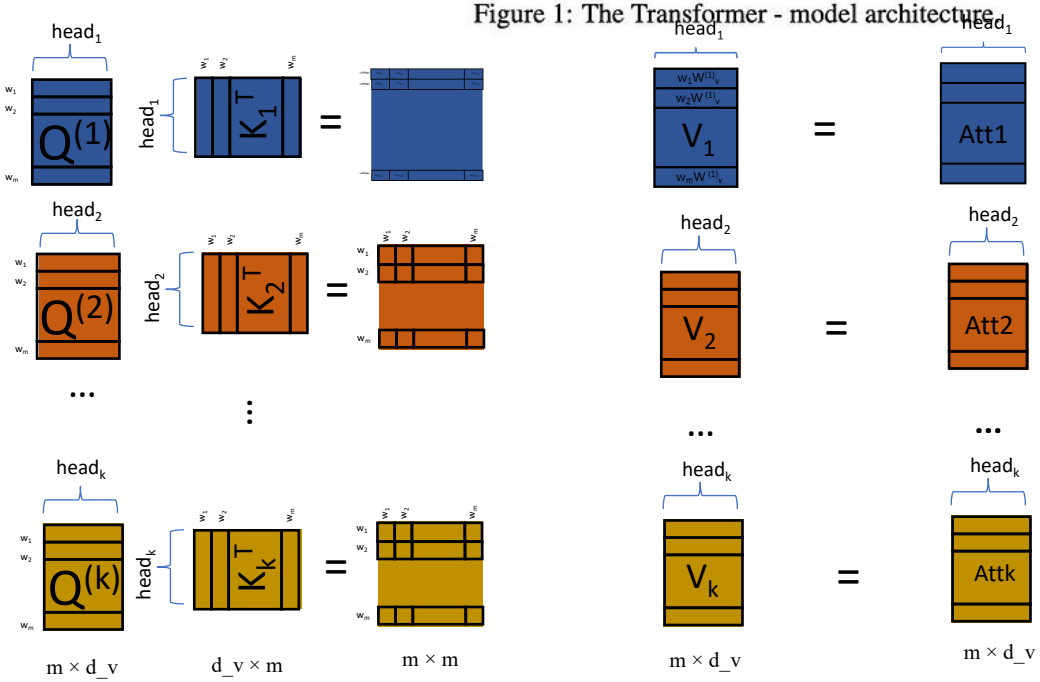
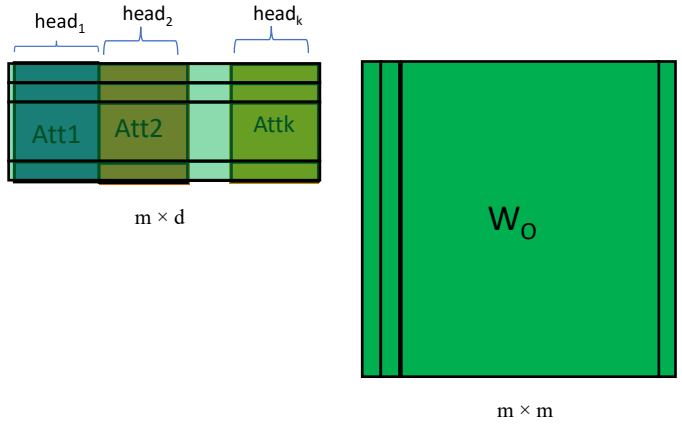


Figure 1: The Transformer - model architecture.





d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k
d_ff >> d

[d = d_model]

Exploring **multi-head attention**

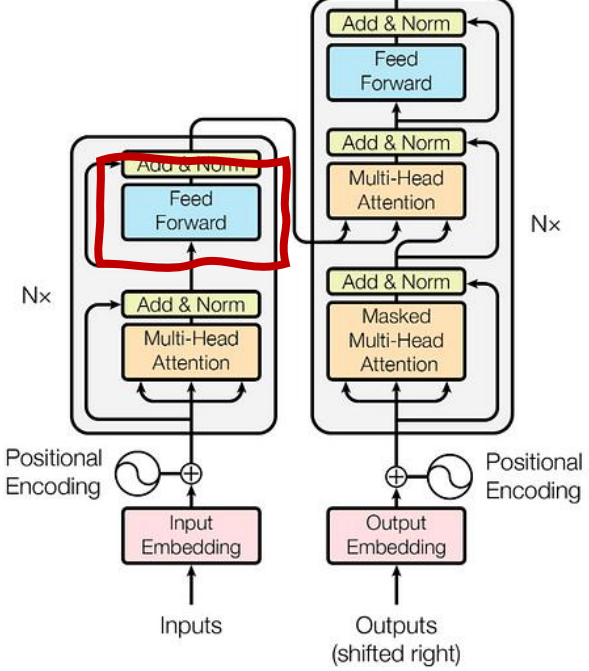
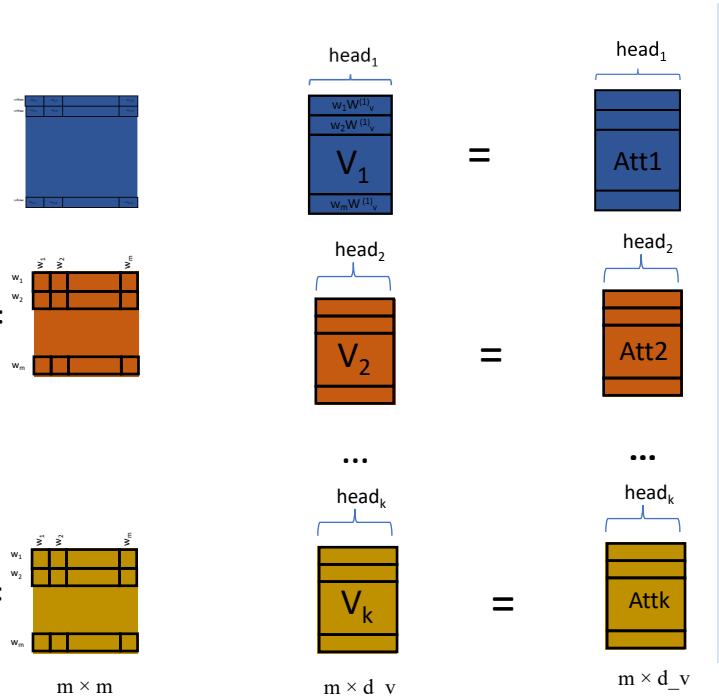
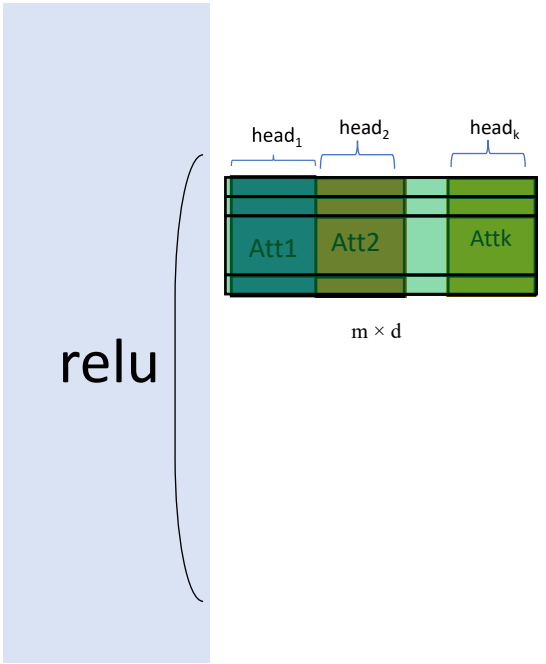
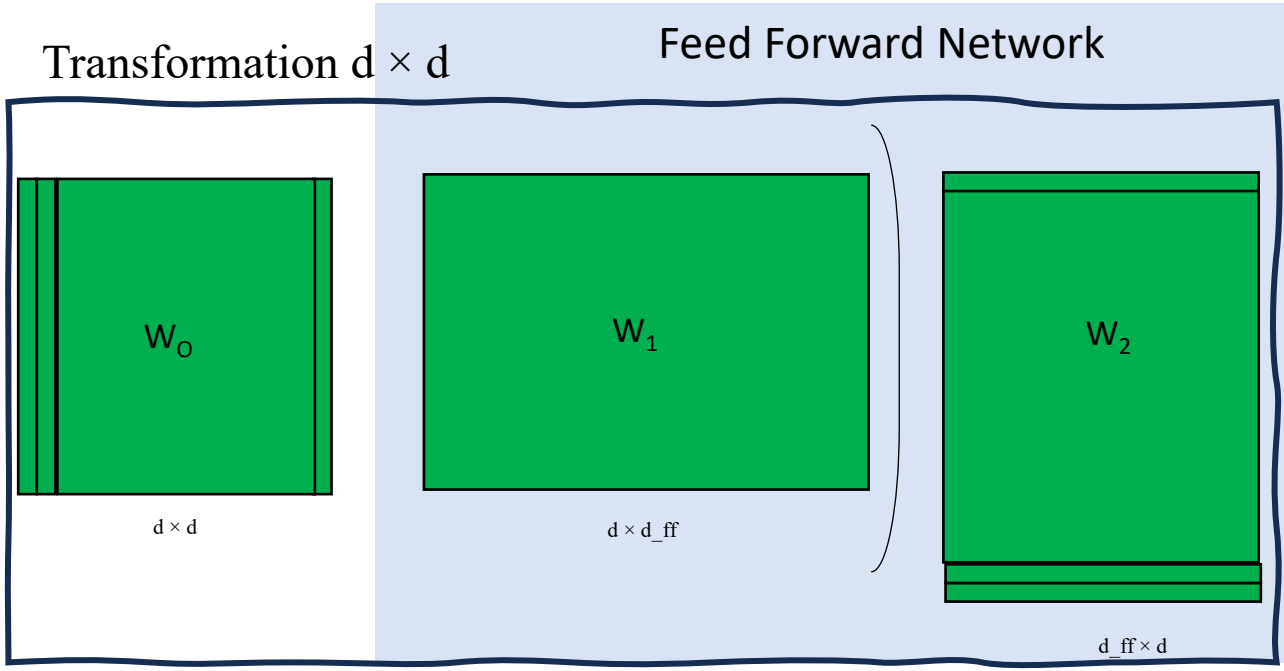


Figure 1: The Transformer - model architecture.



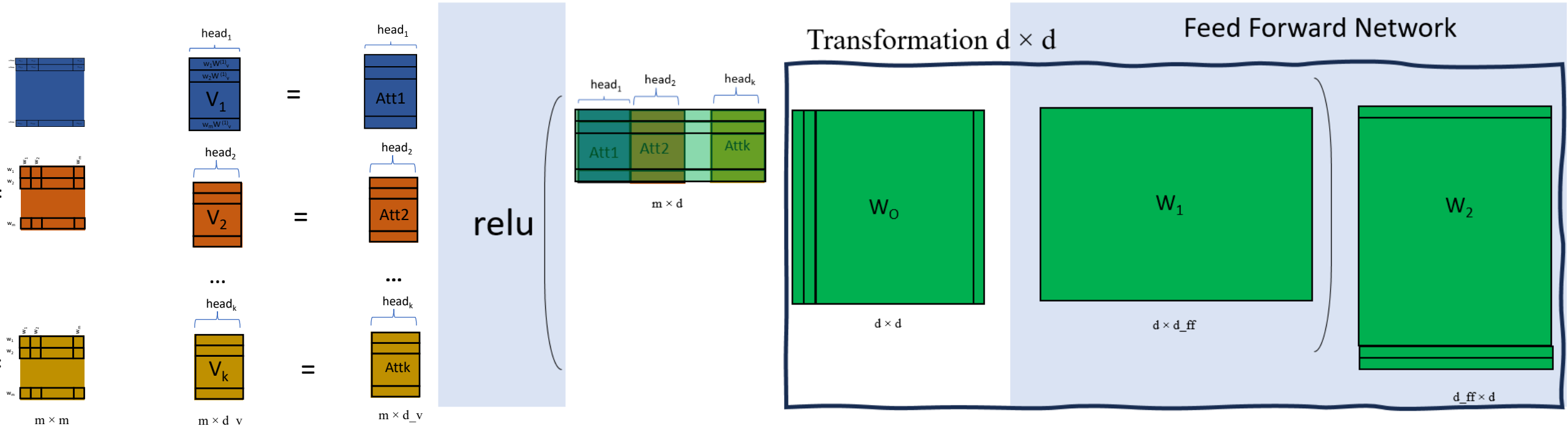


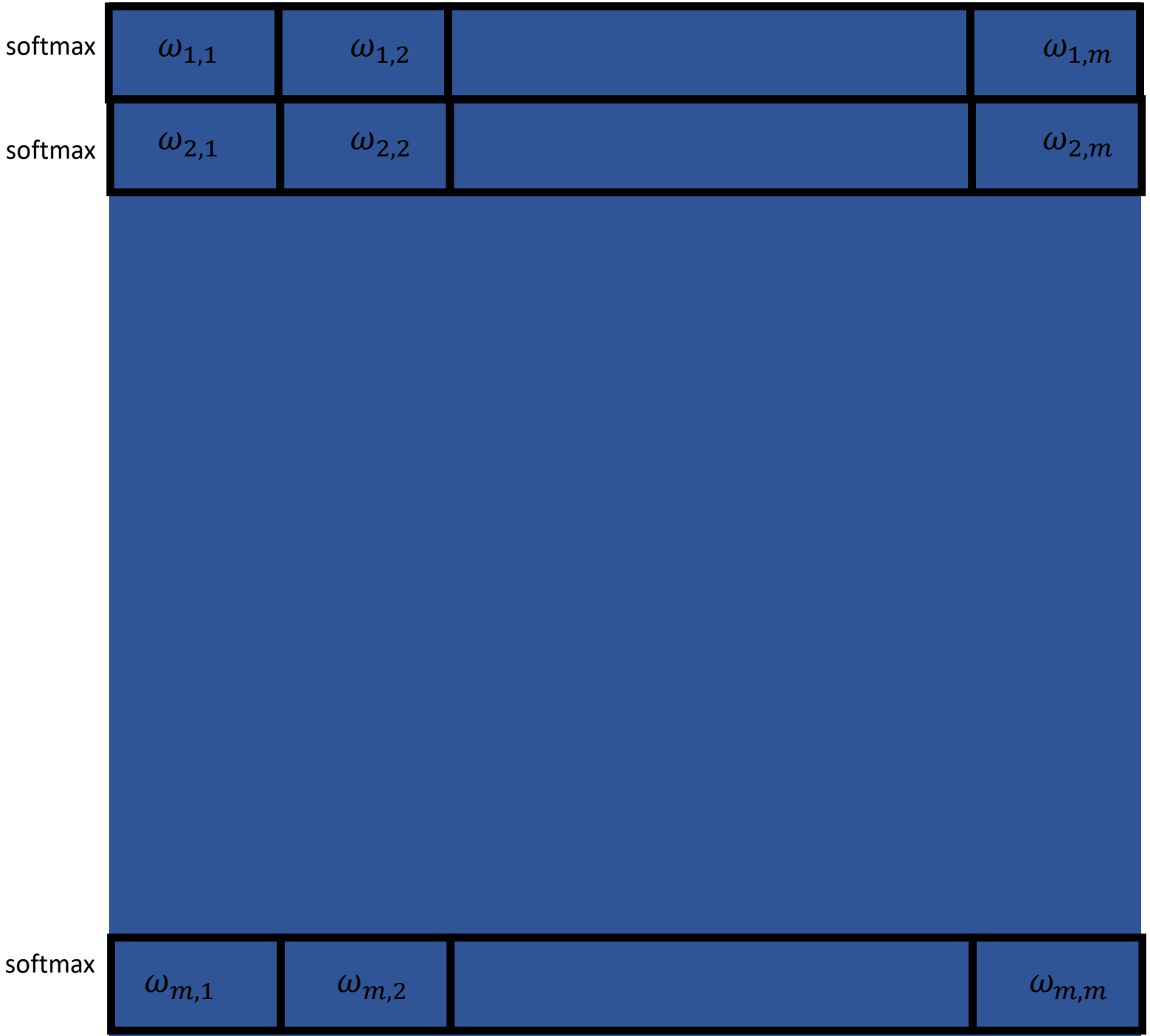


d: internal dimension (embedding)
m: sequence length
k: #heads
d_v = d/k
d_ff >> d

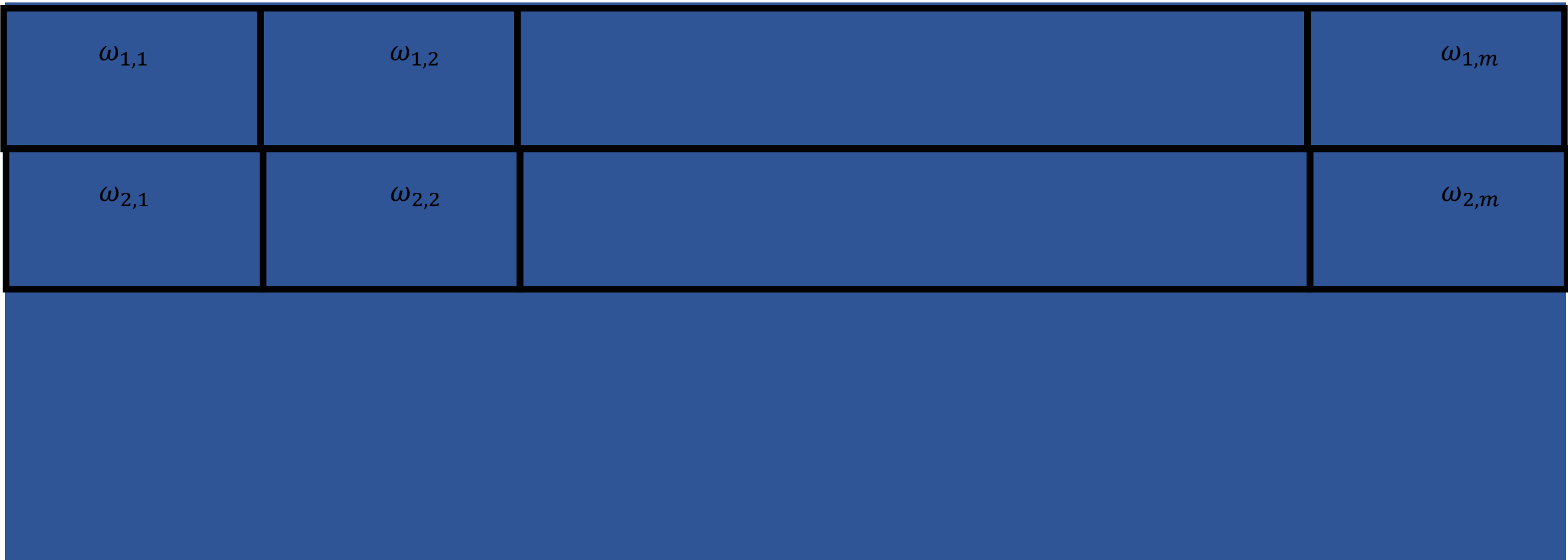
[d = d_model]

Exploring **multi-head attention**



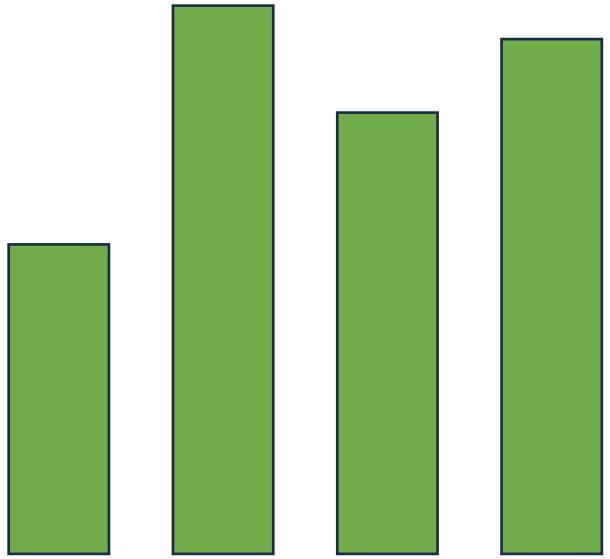

softmax
$\omega_{1,1}$
$\omega_{1,2}$
$\omega_{1,m}$
softmax
$\omega_{2,1}$
$\omega_{2,2}$
$\omega_{2,m}$
softmax
$\omega_{m,1}$
$\omega_{m,2}$
$\omega_{m,m}$

softmax

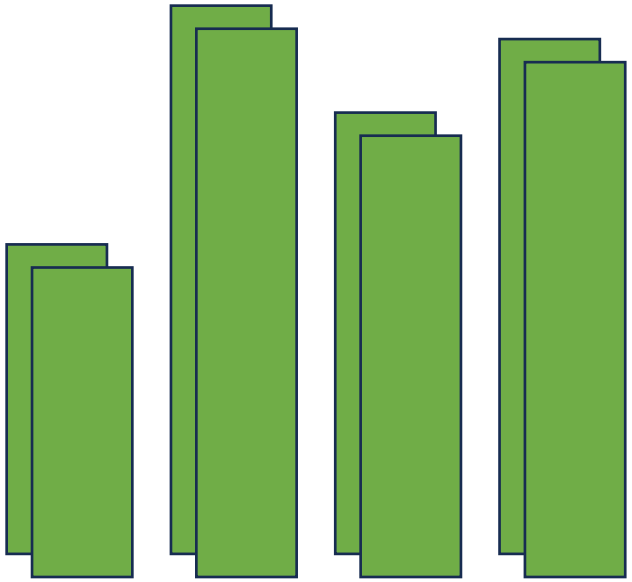


softmax

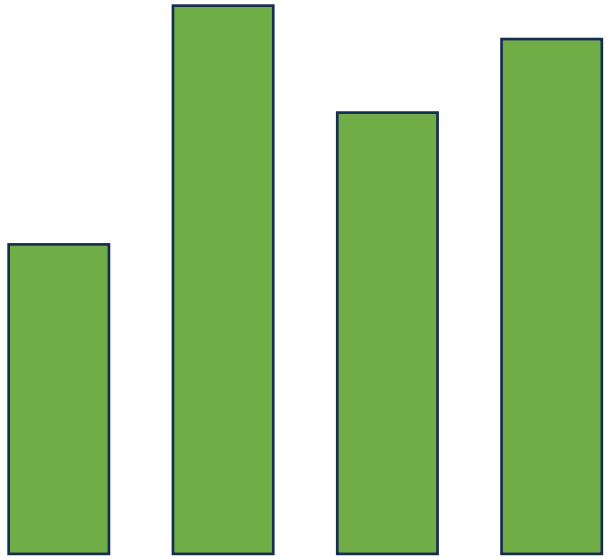
$\omega_{1,1} = w_1 W_K^{(1)} W_Q^{(1)T} w_1^T$	$\omega_{1,2} = w_1 W_K^{(1)} W_Q^{(1)T} w_2^T$		$\omega_{1,m} = w_1 W_K^{(1)} W_Q^{(1)T} w_m^T$
$\omega_{2,1}$	$\omega_{2,2}$		$\omega_{2,m}$



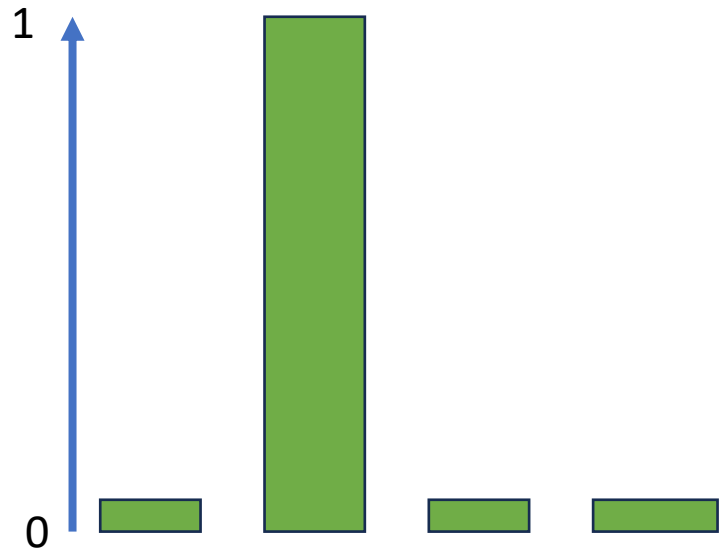
softmax

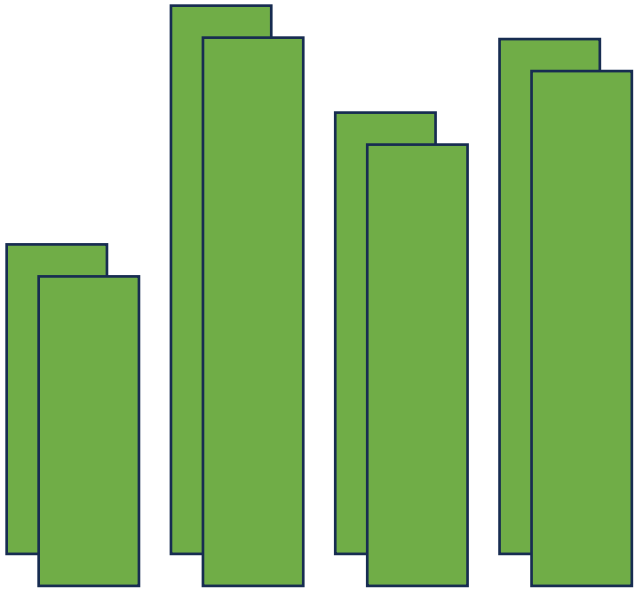


softmax

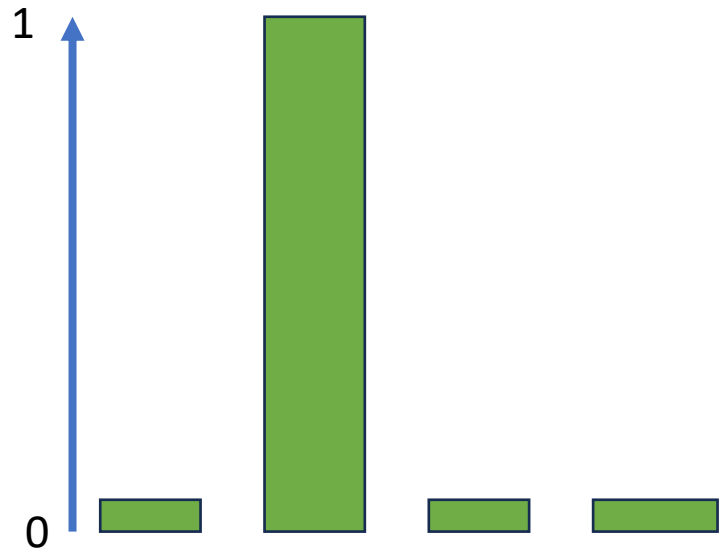


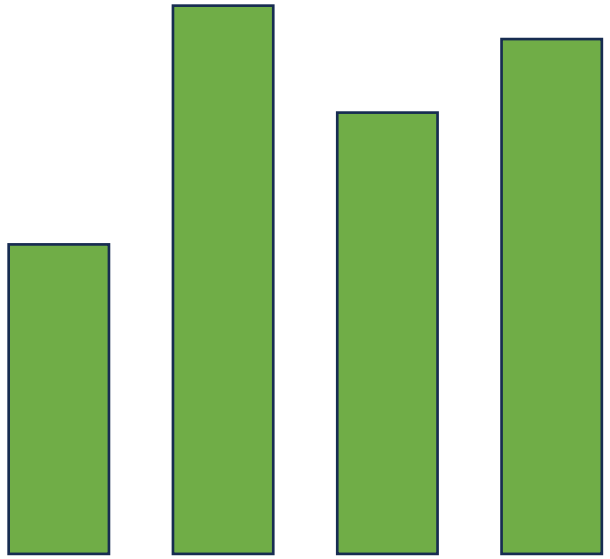
softmax



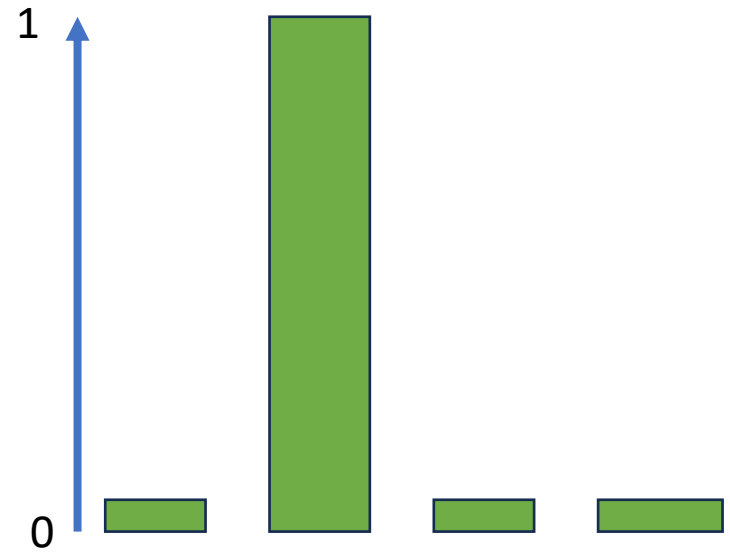


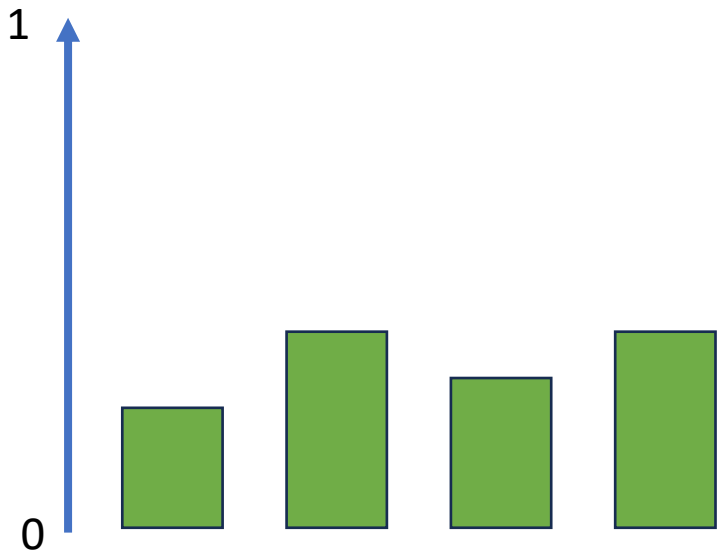
softmax

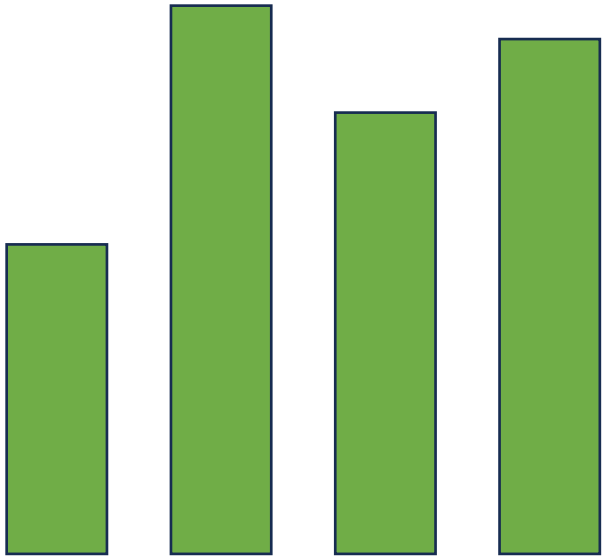




softmax

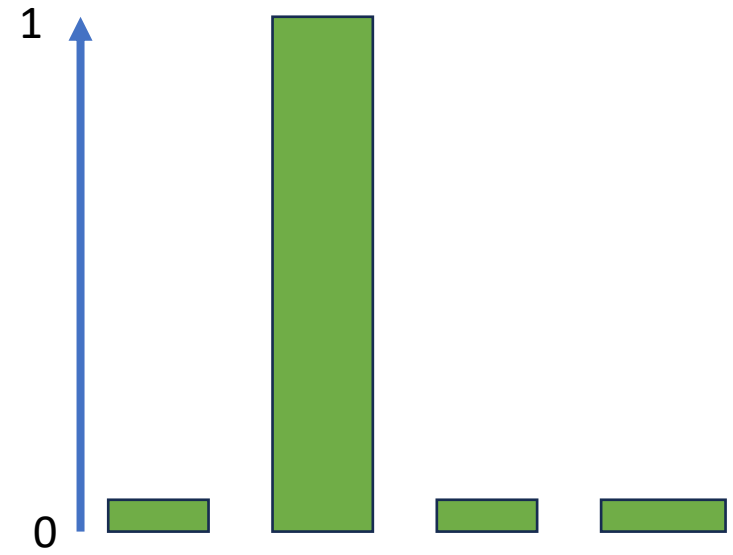






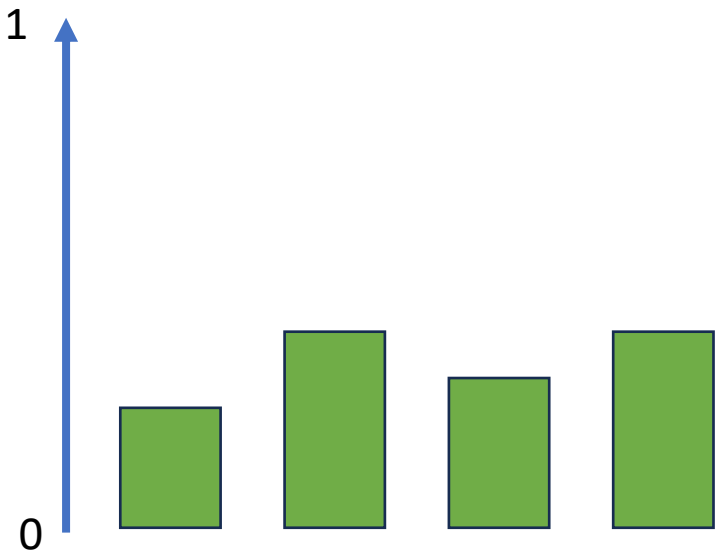
softmax

A)





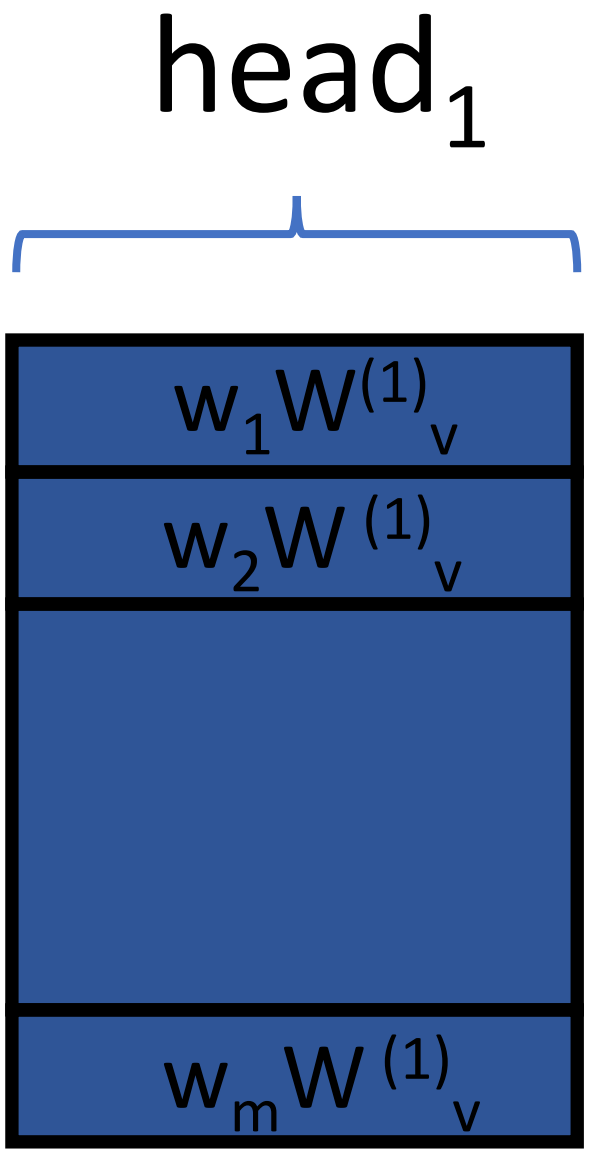
B)

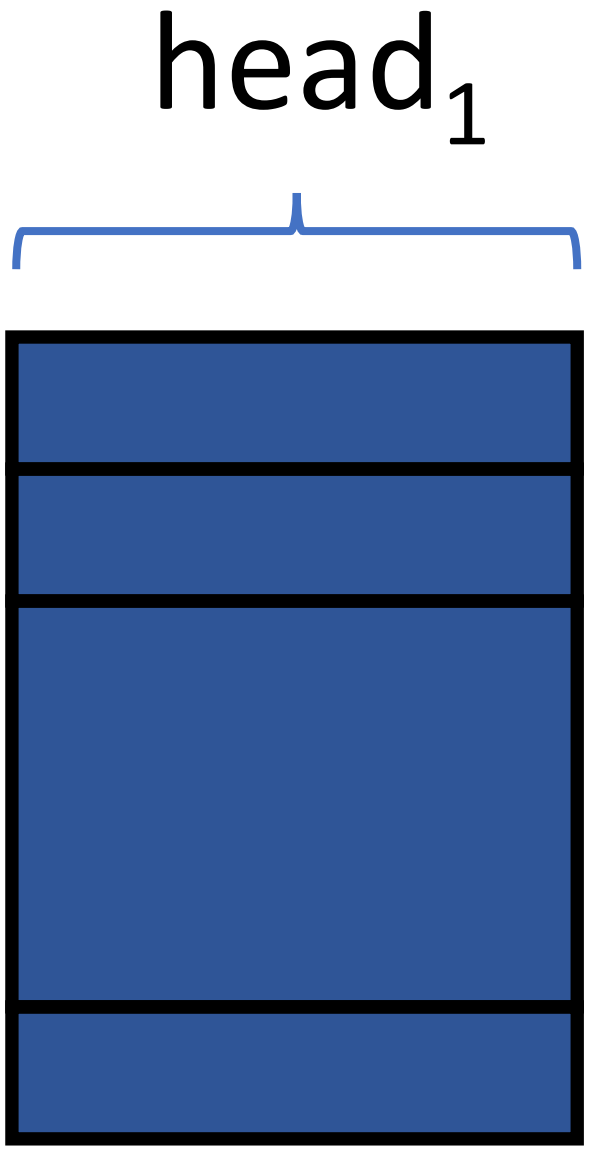


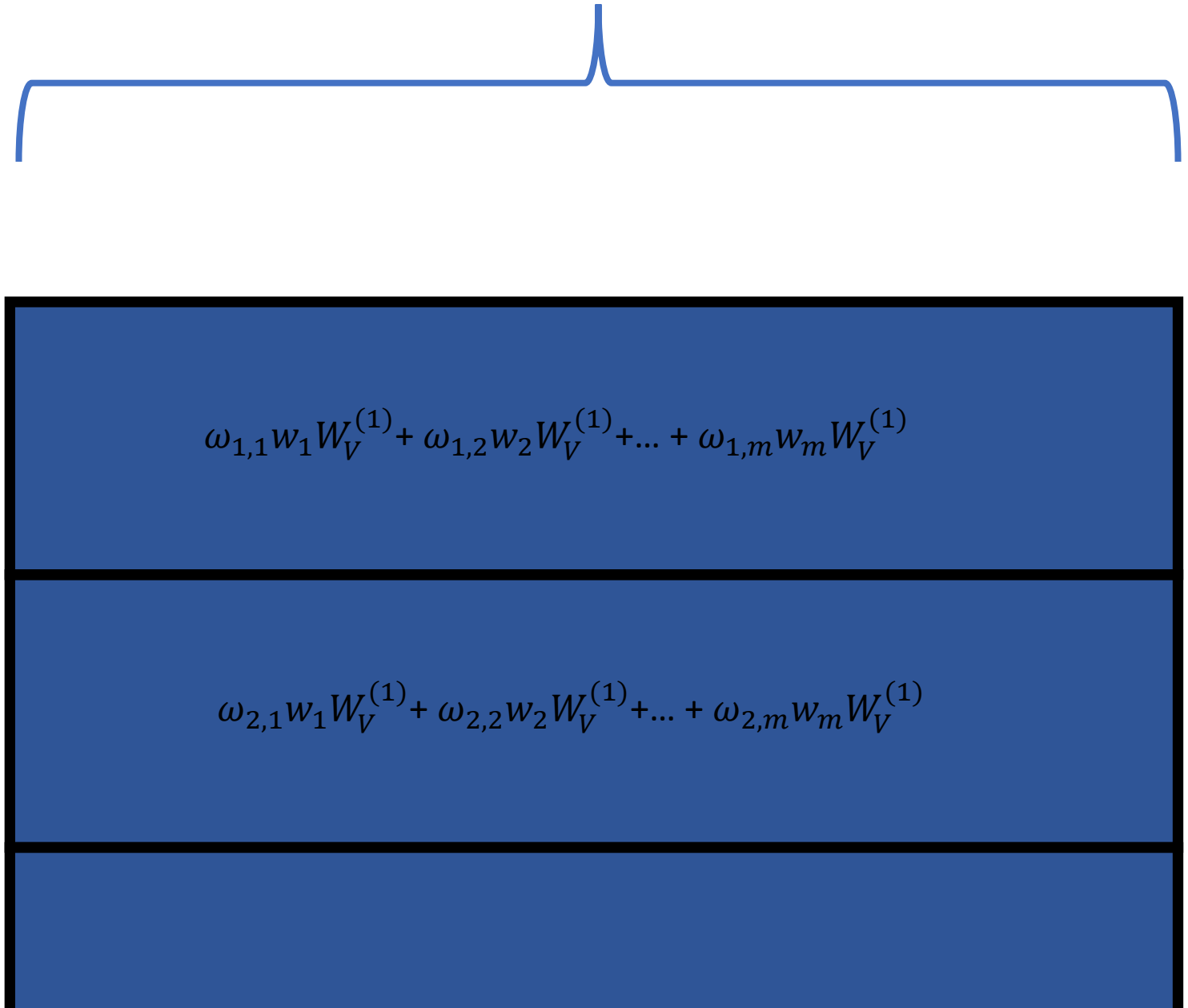
Which one is correct?

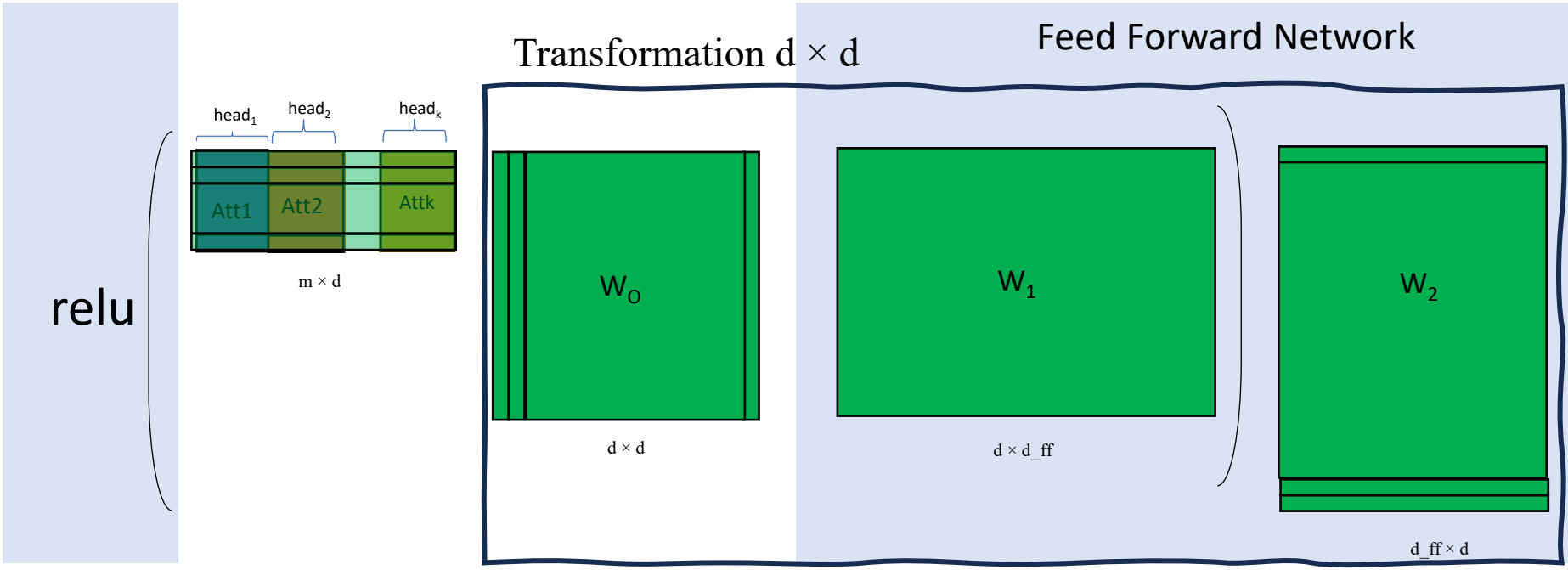


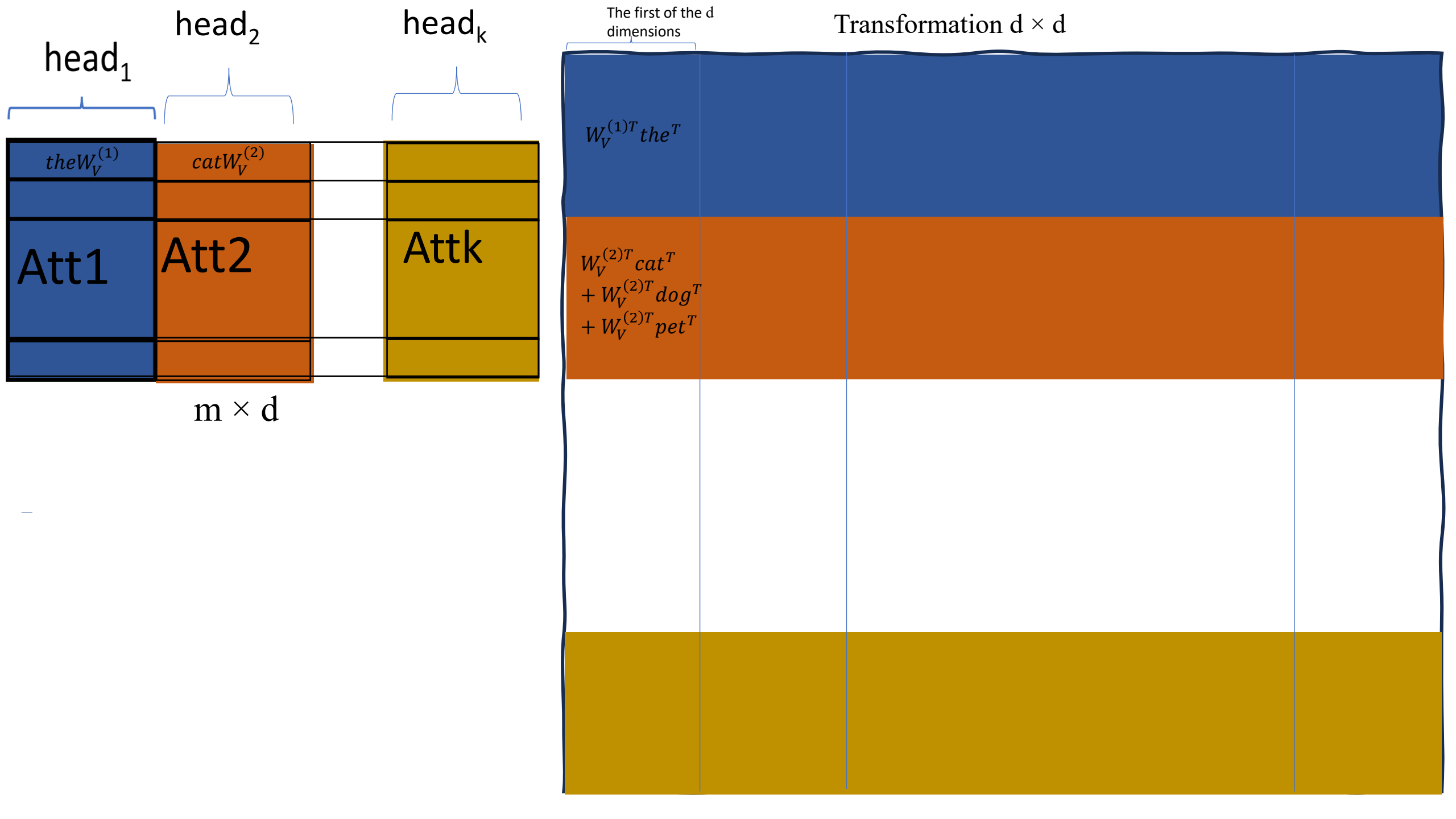


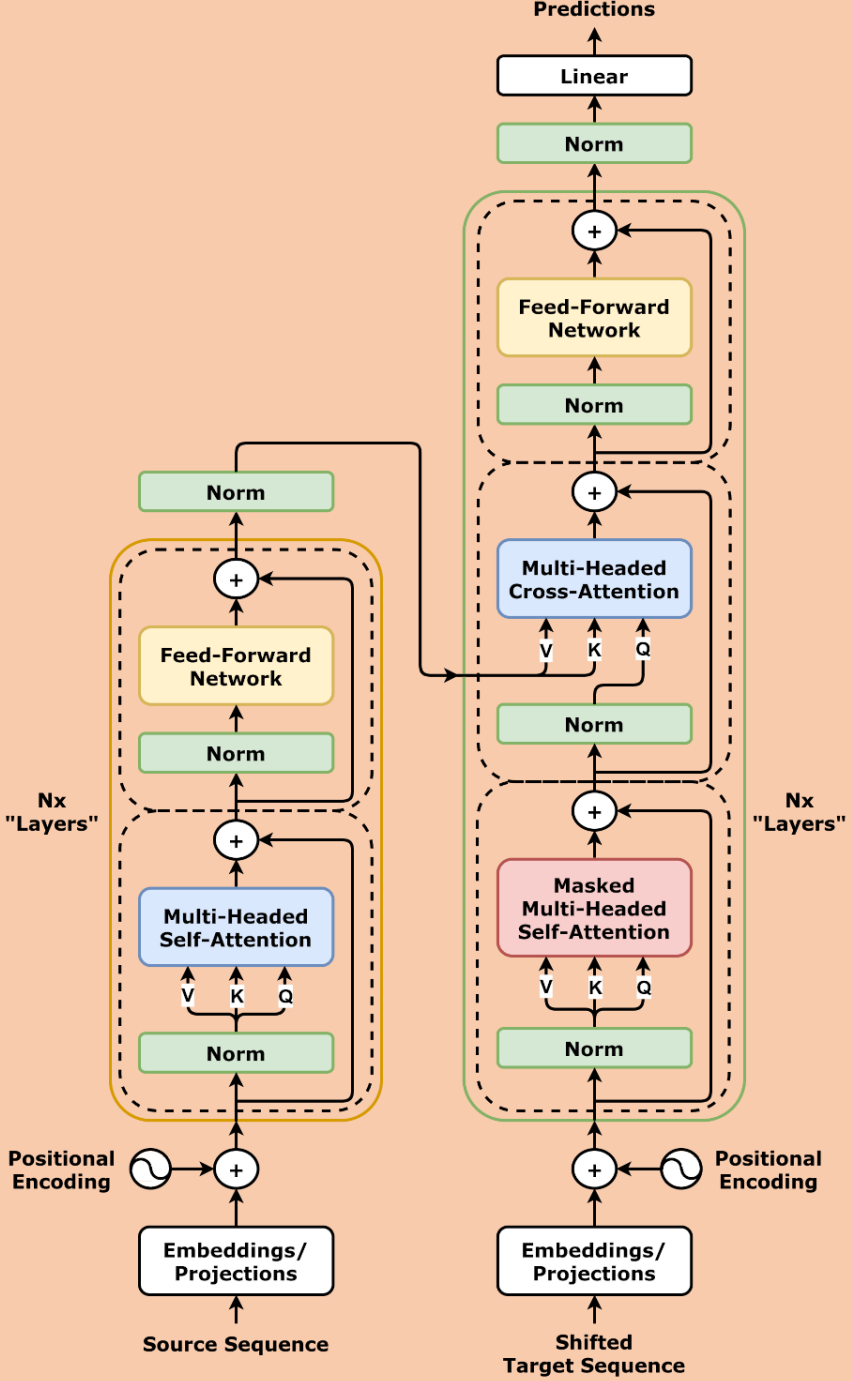

head1
w1W(1)v
w2W(1)v
wmW(1)v


$head_1$


$\omega_{1,1}w_1W_V^{(1)}+\omega_{1,2}w_2W_V^{(1)}+\ldots+\omega_{1,m}w_mW_V^{(1)}$
$\omega_{2,1}w_1W_V^{(1)}+\omega_{2,2}w_2W_V^{(1)}+\ldots+\omega_{2,m}w_mW_V^{(1)}$


relu
head1
head2
headk
Att1
Att2
Attk
m × d
Transformation d × d
Feed Forward Network
W0
d × d
W1
d × d_ff
W2
d_ff × d


head1
head2
headk
theW_V^(1)
catW_V^(2)
Att1
Att2
Attk
m × d
The first of the d dimensions
Transformation d × d
W_V^(1)T the^T
W_V^(2)T cat^T
+ W_V^(2)T dog^T
+ W_V^(2)T pet^T



$W_2 =$

$W_1 =$

$W_0 =$

$W_K =$

$W_Q =$

$W_V =$

Feed-forward seems to memorize (possibly generalized) patterns

Feed-forward seems to memorize (possibly generalized) patterns

Behavior of LLMs

What are the elements that are responsible for a specific prediction?

Feed-forward seems to memorize (possibly generalized) patterns

Behavior of LLMs

What are the elements that are responsible for a specific prediction?

The Role of the Attention Mechanism

The Feed Forward Module

Editing Memories

The Role of the Attention Mechanism



Attention for context mixing

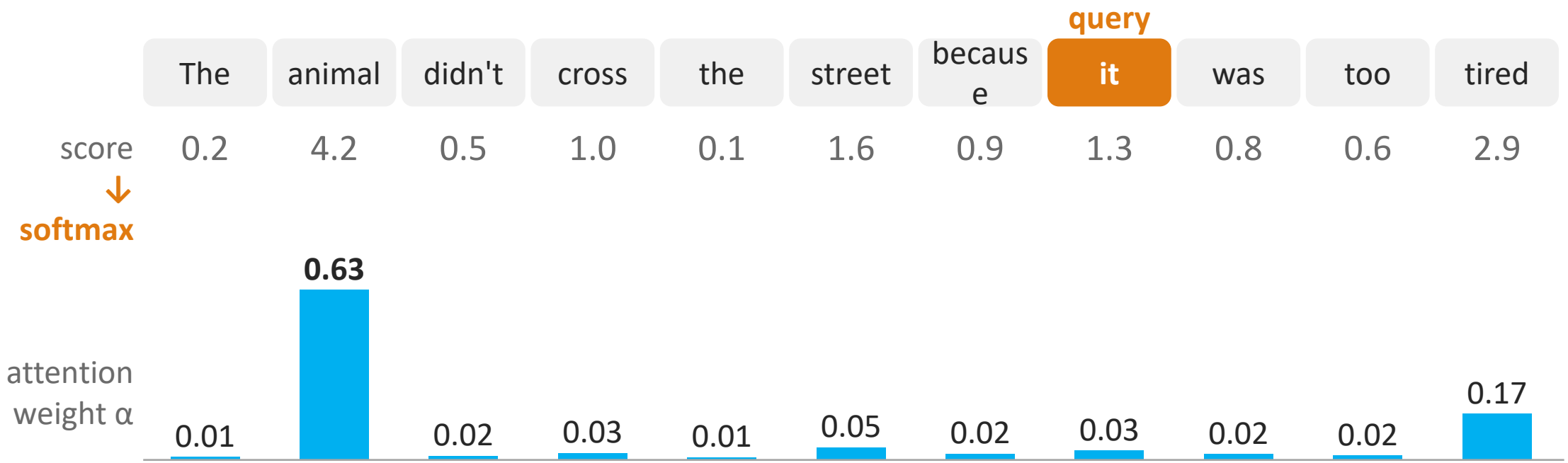
Why Attention?

The “building block” of Transformer based models

“It is naturally interpretable because an attention weight has a clear meaning: how much a particular word will be weighted when computing the next representation for the current word.”

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT’s Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Attention as a weight: toy example



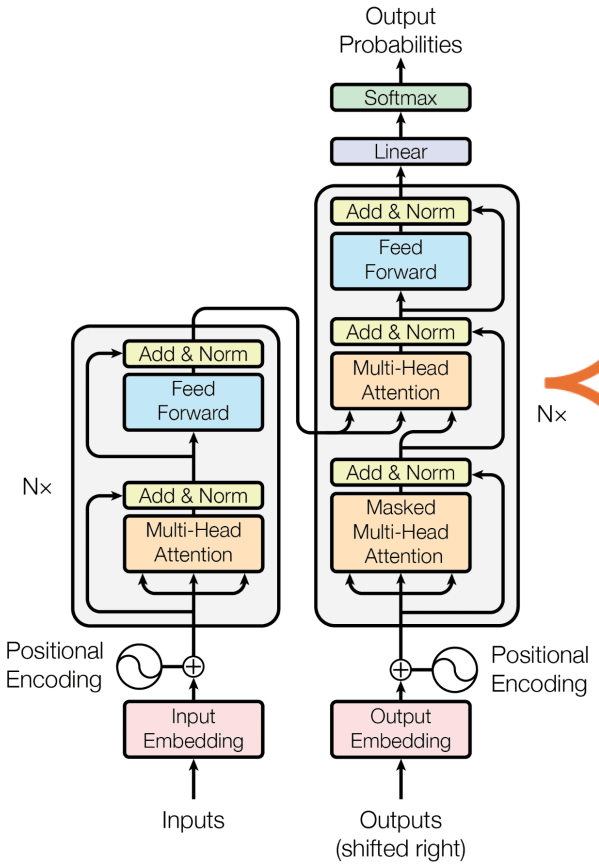
To build the representation of “it”, the model scores every surrounding word.

Attention weights form a probability distribution over the context: non-negative, summing to 1.

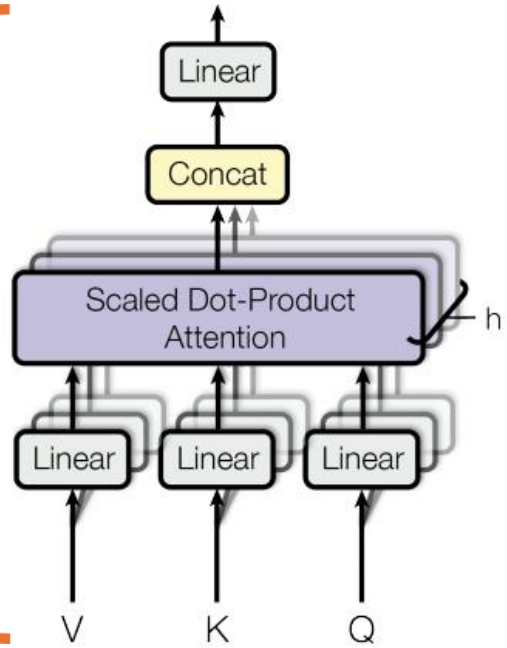
Illustrative scores, hand-made for explanation (not extracted from a model).

Where do scores come from?

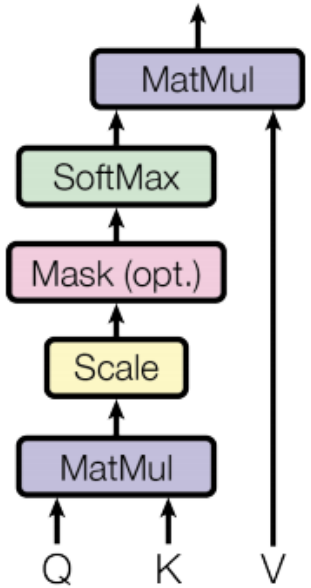
Transformer layer =
Attention block +
Feed Forward



Attention block =
K, V, Q projection matrices +
h **Attention heads**
+ Linear Projection

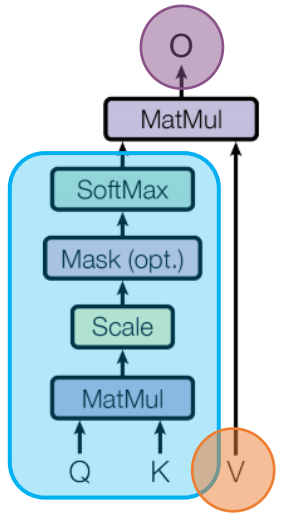


Attention head =
sum of V vectors weighted by
the **attention score**



Attention Head computation

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



Score and softmax

$$\alpha_{ij} = \frac{\exp(q_i^T k_j)}{\sum_{l=1}^{n} \exp(q_i^T k_l)}$$

Weighted sum

$$o_i = \sum_{j=1}^{n} \alpha_{ij} v_j$$

Token embeddings
x_i
for every token

→

Linear projections
$q_i = x_i W^Q$
$k_j = x_j W^K$
$v_j = x_j W^V$

→

Score
$s_{ij} = q_i \cdot k_j$
(scaled by √d)

→

Softmax
α_{ij}
normalized over j

→

Weighted sum
$y_i = \Sigma_j \alpha_{ij} v_j$

Attention Head computation

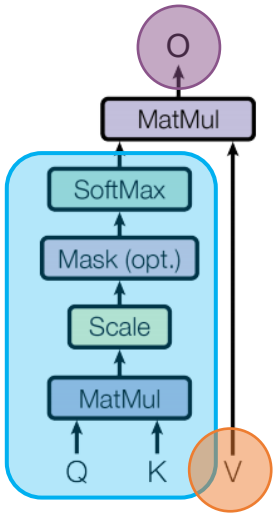
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Score and softmax

$$\alpha_{ij} = \frac{\exp(q_i^T k_j)}{\sum_{l=1}^{n} \exp(q_i^T k_l)}$$

Weighted sum

$$o_i = \sum_{j=1}^{n} \alpha_{ij} v_j$$



- The self-attention mechanism computes a **similarity** between (projected) token representation.
- Given a sequence of length n, the result of this computation is a n x n matrix that could be interpreted as how much a token in a certain position is influencing the representation for an other
- In fact, the attention weight is governing how the token representations projected by the value weight matrix should be mixed to get the representation for each of the tokens

The initial interpretability hypothesis



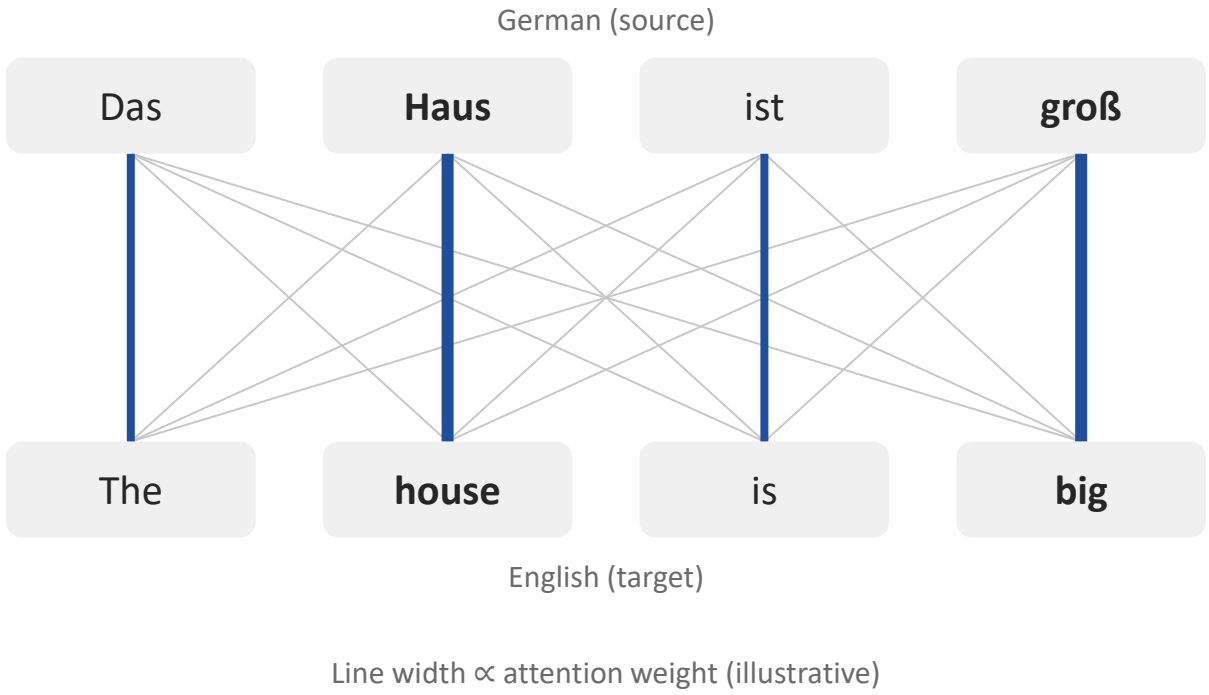
The earliest interpretability work often assumed that attention weights reveal which tokens the model is using.

Input	Attention pattern
John gave Mary a gift	“gift” attends to “gave”
The dog barks	“barks” attends to “dog”

If this interpretation is correct, attention maps should reveal meaningful linguistic structure.

Examples are illustrative.

Attention as soft alignment

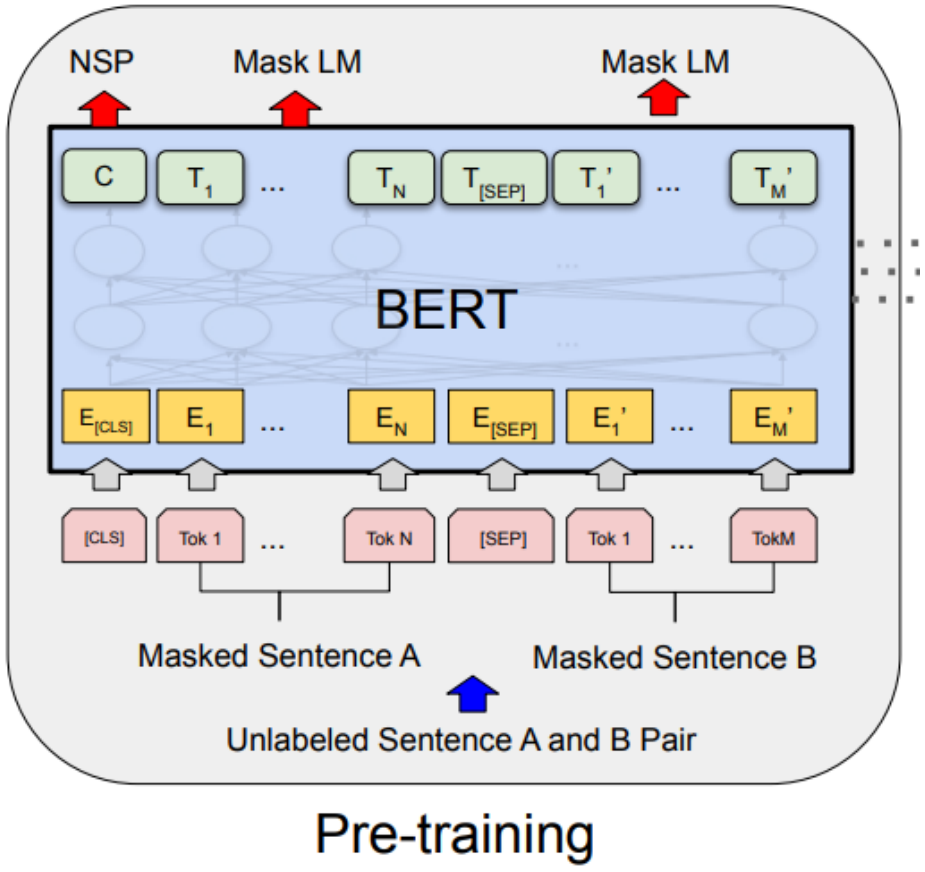


Attention was quickly interpreted as a form of soft word alignment.

This led researchers to ask whether attention heads learn syntax, semantics, and translation alignments.

Illustrative weights, hand-made for explanation.

Attention as soft alignment



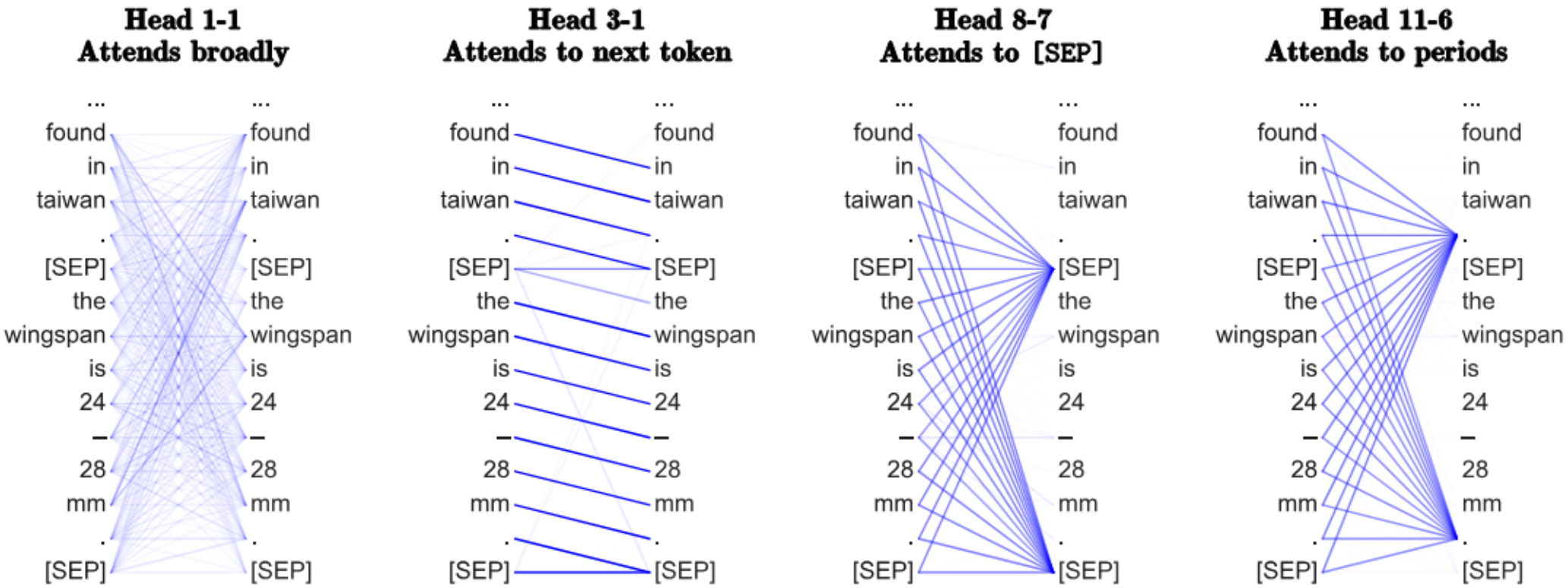
Pre-training

Questions:

- How do heads behave?
  - Do they **attend broadly** to the sentence?
  - Do they **focus** on particular words?
- Without any supervision, do attention heads capture peculiar **linguistic phenomena?**

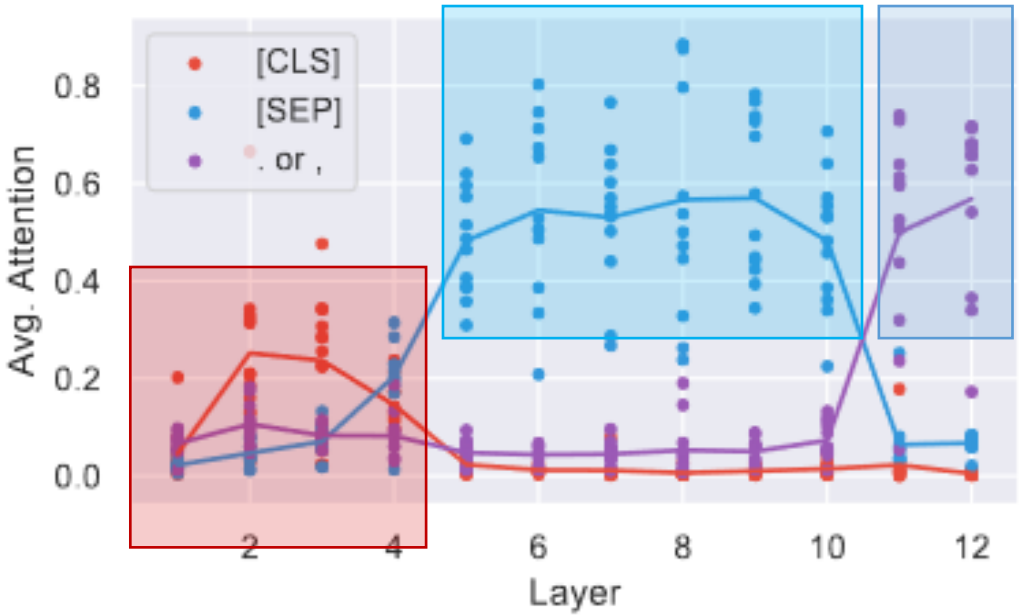
Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Surface Level Patterns in Attention



Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

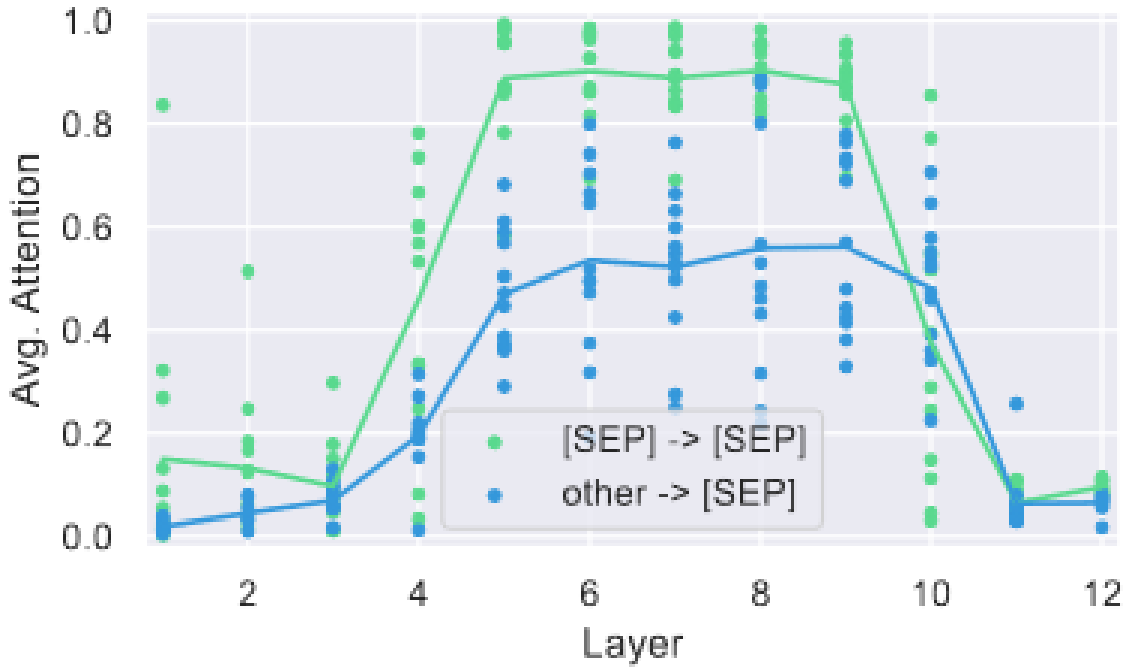
High attention weights to semantically empty tokens



- Each point corresponds to the average attention weight for a head
- A substantial amount of BERT's attention focuses on a **few** tokens
  - Heads often attend to "special" tokens
  - For example, over half of BERT's attention in layers 6-10 focuses on [SEP] (baseline 1/64 since [SEP] appears twice in their 128 tokens examples)

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Is [SEP] aggregating information?

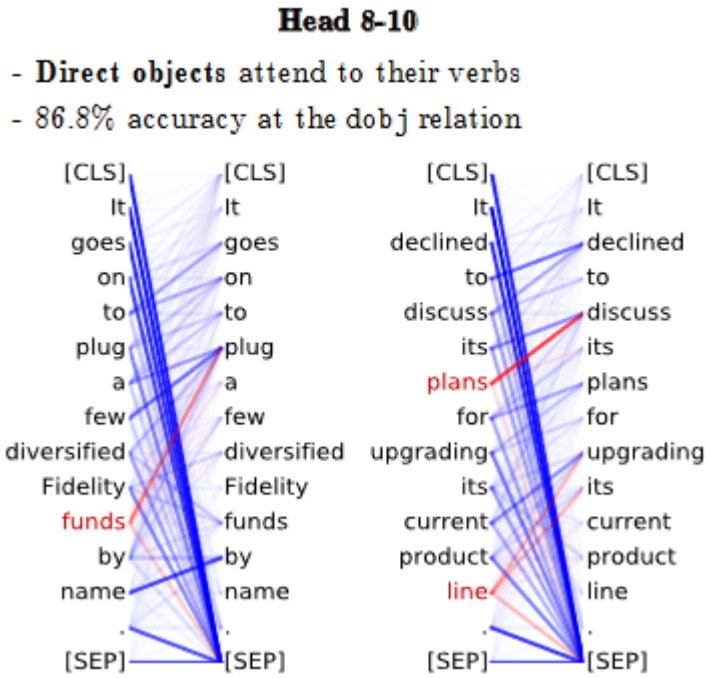


- If that was true, we would expect attention heads processing [SEP] to attend broadly over the whole segment to build up these representations.
- However, they instead almost entirely (more than 90%) attend to themselves and the other [SEP] token

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Some heads with specific functions

Direct objects focus on the verbs

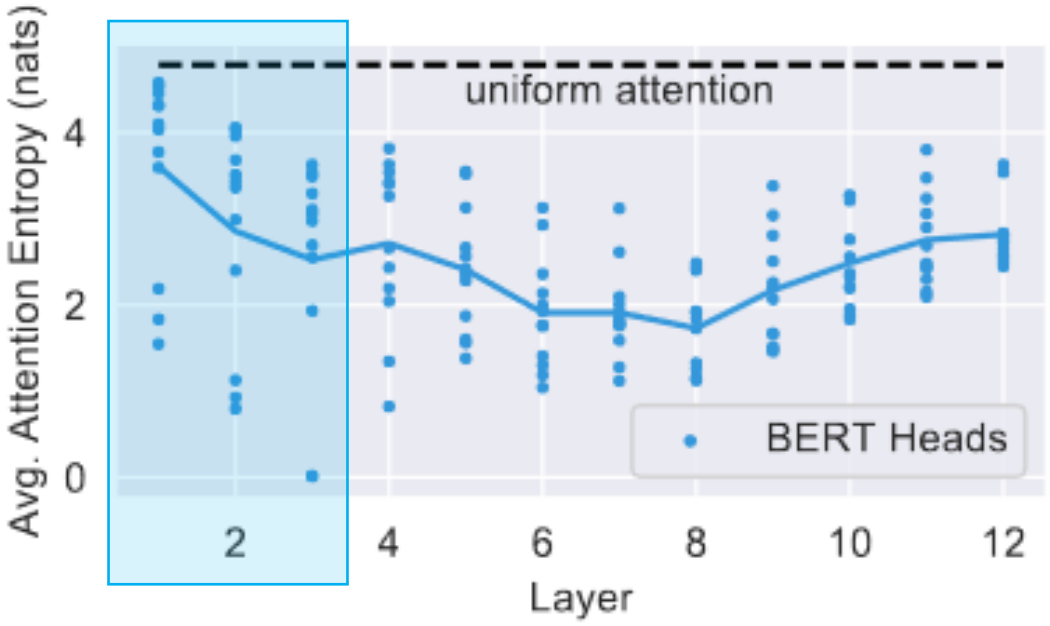


Focus on previous token in early layers



Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

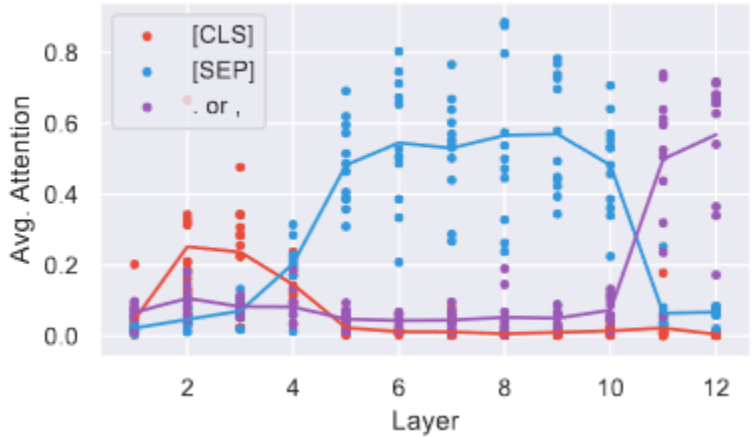
Attention can be very spread

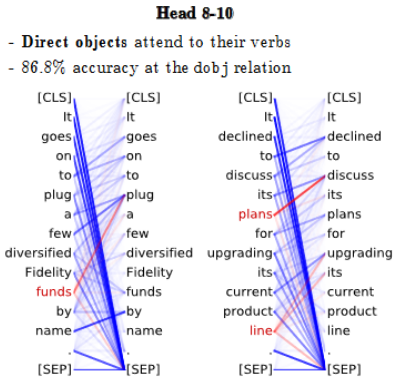


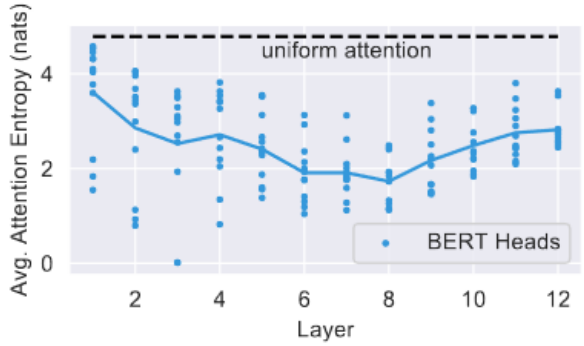
- Compute the average entropy of each head's attention distribution
- In the first layer there are particularly high-entropy heads that produce bag-of-vector-like representations.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

So far, Attention does **not** have clear interpretation



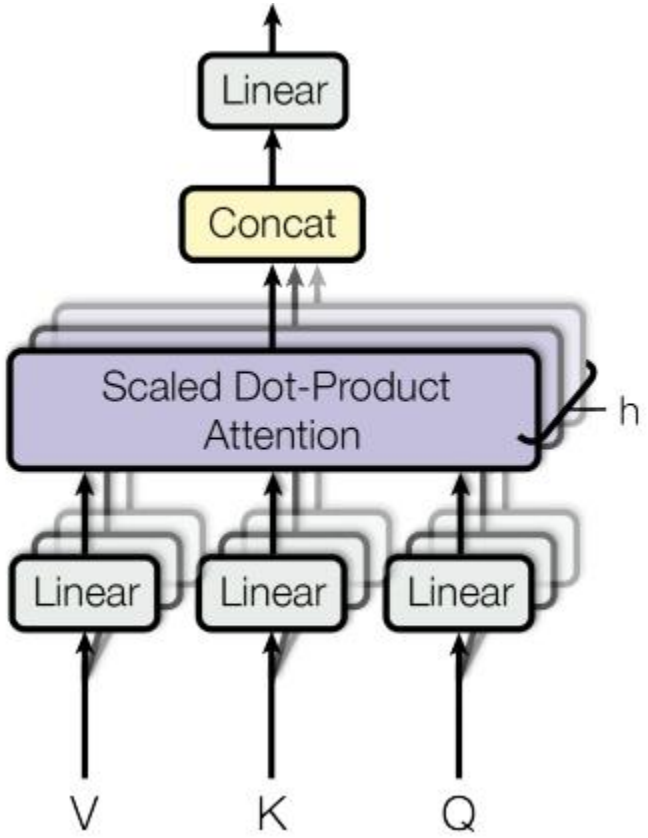




- High attention scores to not "meaningful" tokens
- Some heads seem to encode *syntactic* information
- Overall, the attention weights are quite spread

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

So far, Attention does **not** have clear interpretation



- We are also not able to discuss:
  - What multiple heads are doing
  - What the final linear layer in the attention block is computing

Rewriting Attention

Standard view

$$y_i = (\Sigma_j \alpha_{ij} v_j) W^O$$

$v_j = x_j W^V + b^V$ is the value vector; the head output goes through W^O

Move W^O inside

$$y_i = \Sigma_j \alpha_{ij} (v_j W^O)$$

W^O is a linear map, so it can be applied to each term of the sum

Define f

$$\mathbf{f(x_j)} := (x_j W^V + b^V) W^O \quad \Rightarrow \quad \mathbf{o_i = \Sigma_j \alpha_{ij} f(x_j)}$$

$f(x_j)$ is what token j has to offer, computed from x_j alone

Each input token j contributes one vector, $\alpha_{ij} f(x_j)$. The output o_i is the sum of these n pieces.

α_{ij} depends on the query of token i and the key of token j; $f(x_j)$ depends only on token j.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

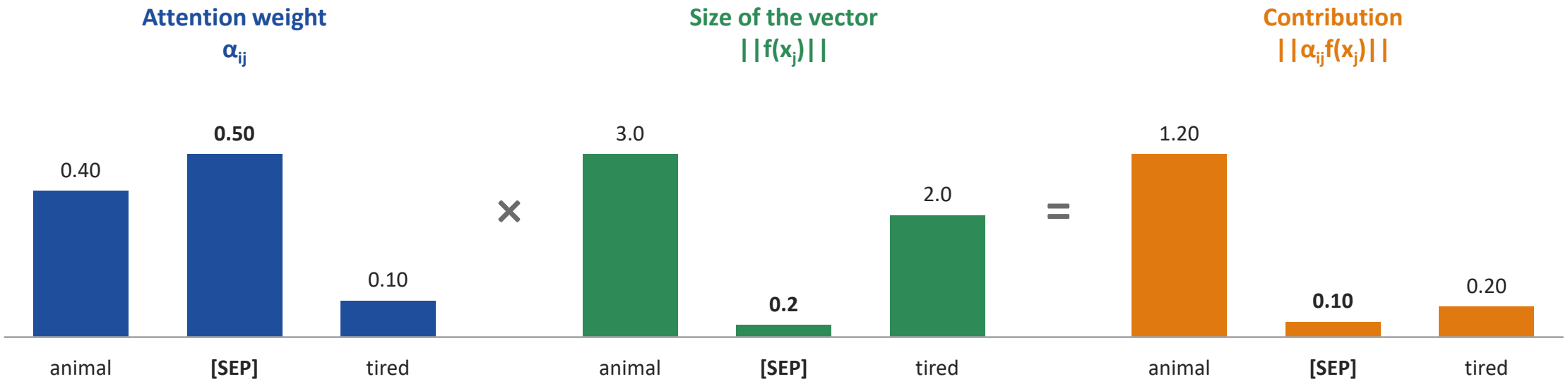
Rewriting Attention

$$f(x_j) := (x_j W^V + b^V) W^O \quad \Rightarrow \quad o_i = \Sigma_j \alpha_{ij} f(x_j)$$

- Rewriting the attention mechanism it is clear that:
  - first transforms each input vector x to generate f(x)
  - computes attention weights α ;
  - and then compute the **sum** αf(x)

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Weights are not the whole story



[SEP] gets the largest weight (0.50) but contributes the least: its vector is almost zero.

Weight-based: look at α_{ij}

Norm-based: look at $||\alpha_{ij} f(x_j)||$

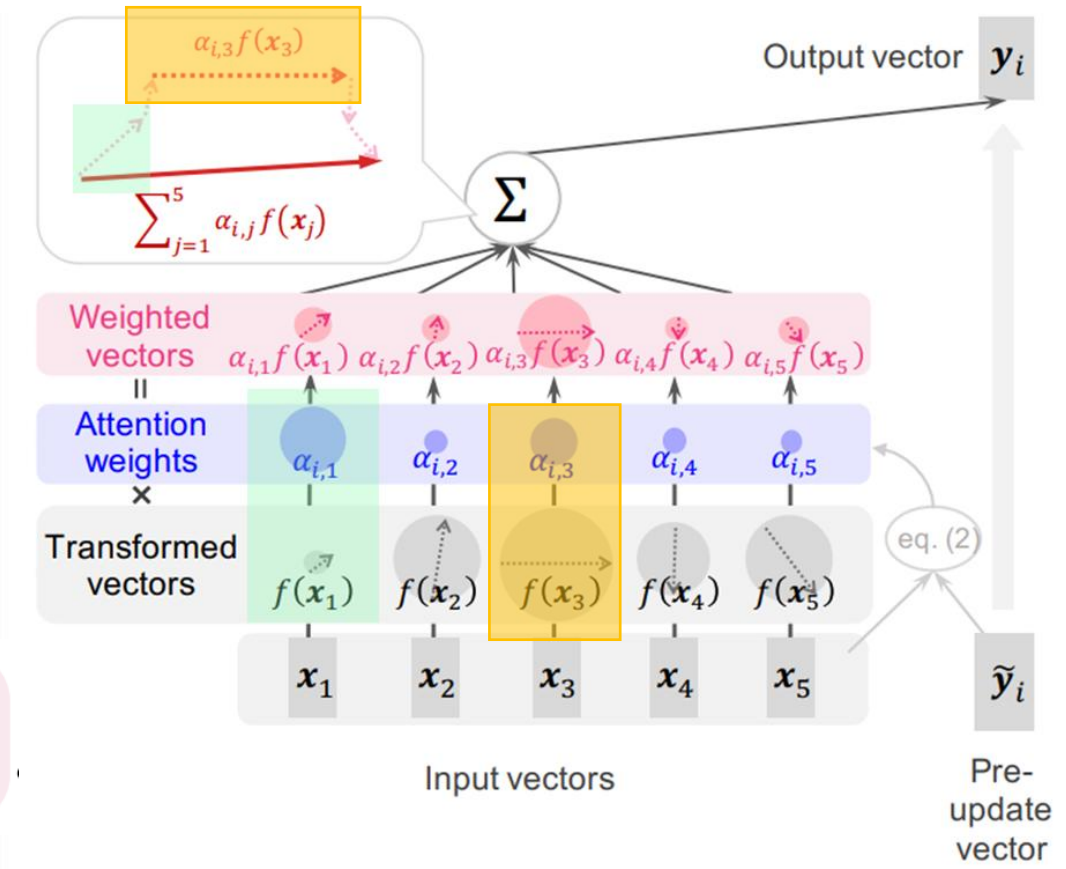
Illustrative values, hand-made for explanation.

Weight-based analysis vs norm-based

- The analysis on the attention weights alone disregard the **magnitude** of the input vectors
- See what happen with large attention weight but small $||f(x_1)||$…
- … and the effect of x_3

$$\boldsymbol{y}_i = \sum_{j=1}^{n} \alpha_{i,j} \, f(\boldsymbol{x}_j)$$

$$\|\alpha f(\boldsymbol{x})\|$$



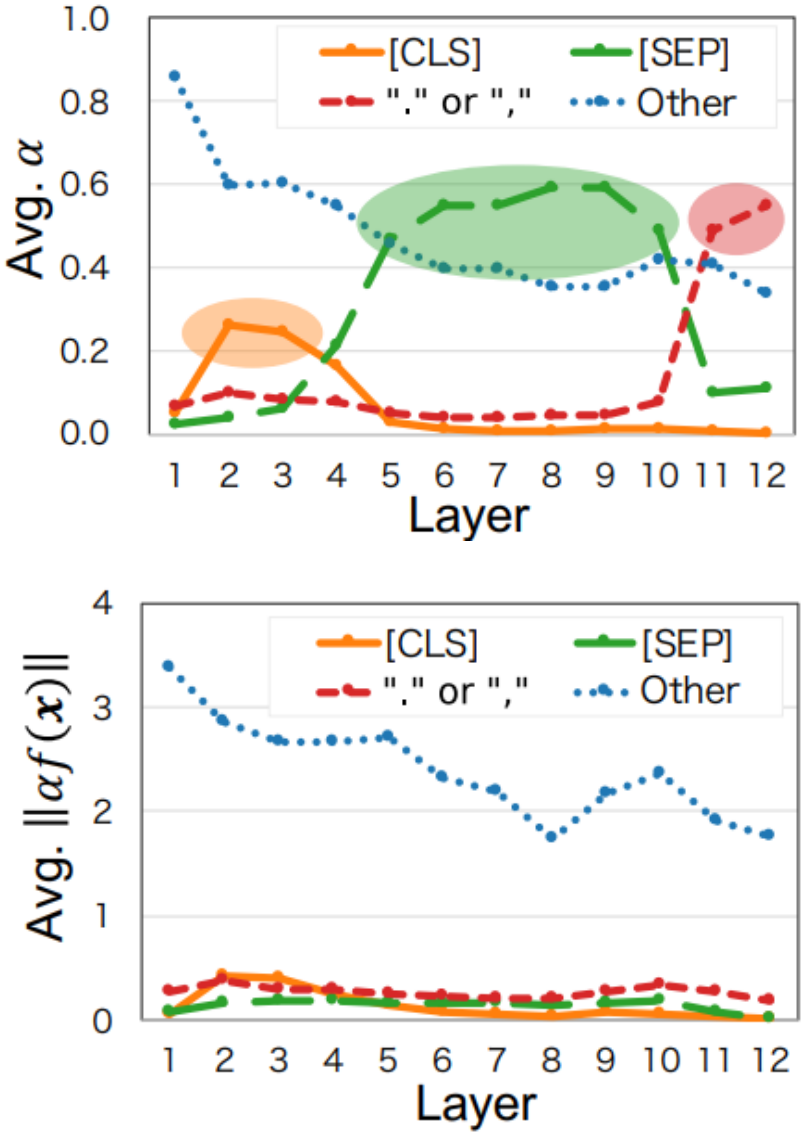
Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Experiments: [SEP] is not important

Replicate analysis on BERT of Clark, 2019.

The weight-based and norm-based analyses exhibited entirely different trend.

Norm-based analysis demonstrates that the amount of information from the vectors corresponding to [SEP] is small

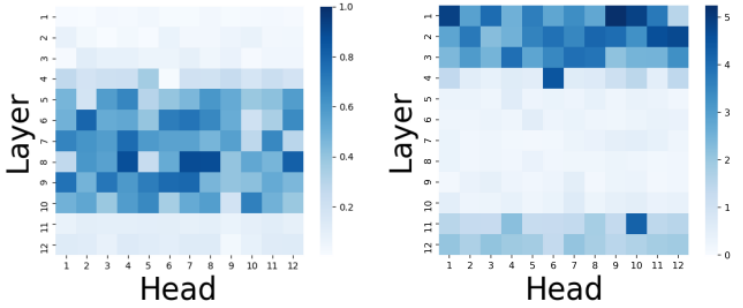


Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Experiments: Attention weights and Norm cancel each other

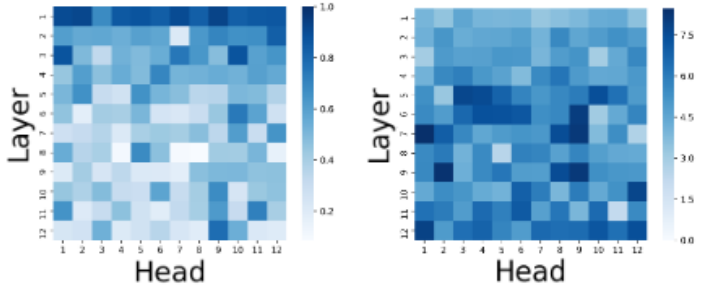
Token category	Number of vectors	Spearman's ρ
[CLS]	17,443,296	-0.34
[SEP]	34,886,592	-0.69
comma & period	182,838,528	-0.25
Others	1,944,928,224	-0.06

Table 2: Spearman rank correlation coefficient between α and $\|f(\boldsymbol{x})\|$ in each token category.



(a) α. (b) $\|f(\boldsymbol{x})\|$.

Figure 4: The higher value of averaged α or $\|f(\boldsymbol{x})\|$ for [SEP] tokens in a given head, the darker its cell.



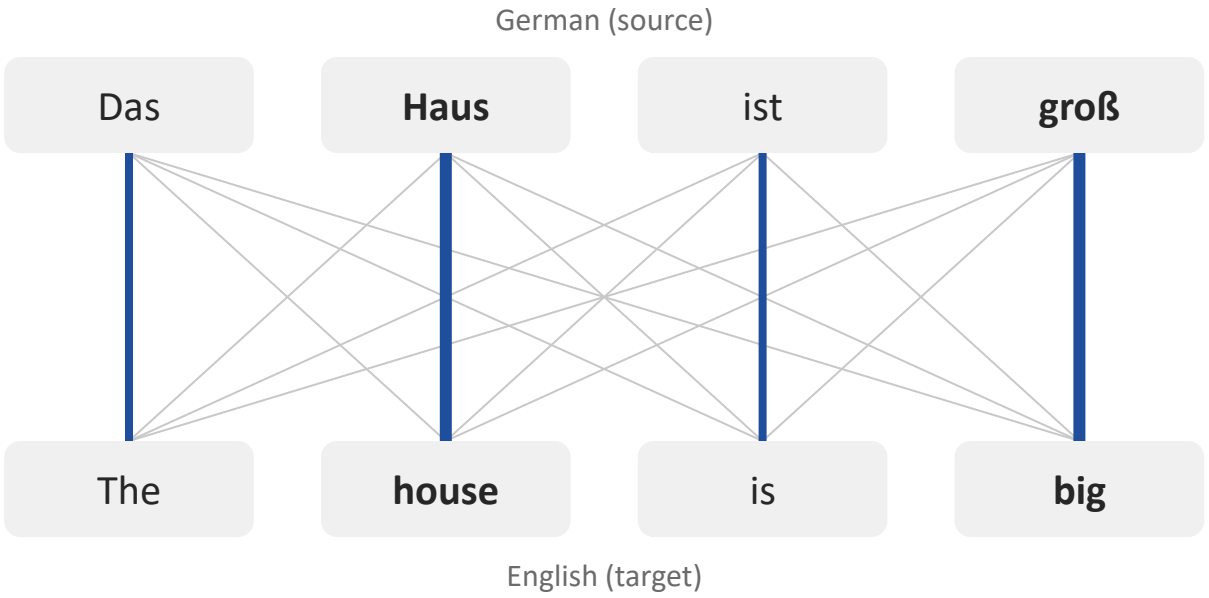
(a) α. (b) $\|f(\boldsymbol{x})\|$.

Figure 12: α and $\|f(\boldsymbol{x})\|$ corresponding to other tokens, averaged on all the input text.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Word Alignment for Neural Machine Translation

- One major research topic in the NMT field is whether NMT systems internally capture word alignment between source and target texts
- **How can word alignment be extracted from black-box NMT systems?**
  - Previous works empirically showed, using the weight-based method, that word alignment induced by the attention of the Transformer is *noisy*.
- The analysis of source-target attention using vector norms and demonstrate that clean alignments can be extracted from the source-target attention.



Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

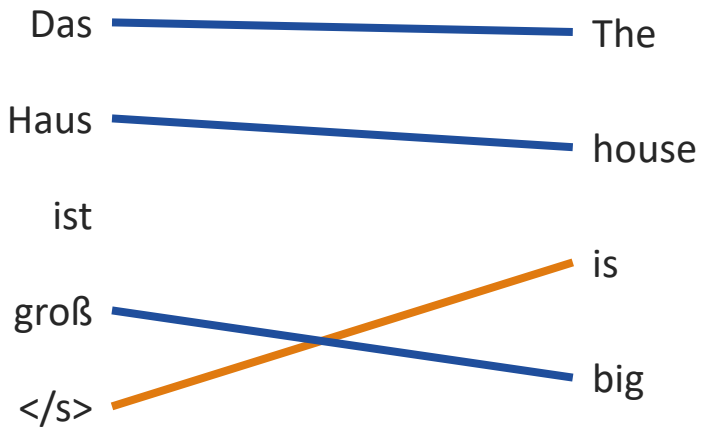
From attention to word alignment

Source–target attention (rows = target, columns = source)

	Das	Haus	ist	groß	</s>
The	.70	.10	.05	.05	.10
house	.10	.70	.05	.05	.10
is	.05	.10	.30	.05	.50
big	.05	.10	.05	.70	.10

argmax per row

Resulting alignment (source → target)



Each target word is linked to the source word with the highest weight.
Illustrative failure: “is” puts most of its weight on </s>, a special token, so its link (orange) is wrong.
The norm-based analysis that follows tries to addresses exactly this.

Illustrative values, hand-made for explanation (not taken from the paper).

Word Alignment for Neural Machine Translation

- Given {s1...sj} a sentence in a language and {t1… ti} the translated version of it
- **Word alignment** is estimated by calculating **a source word sj** that has the **highest weight** when **generating** a target word **ti**
- **Weight-based alignment** and **norm-based alignment** are computed respectively as:

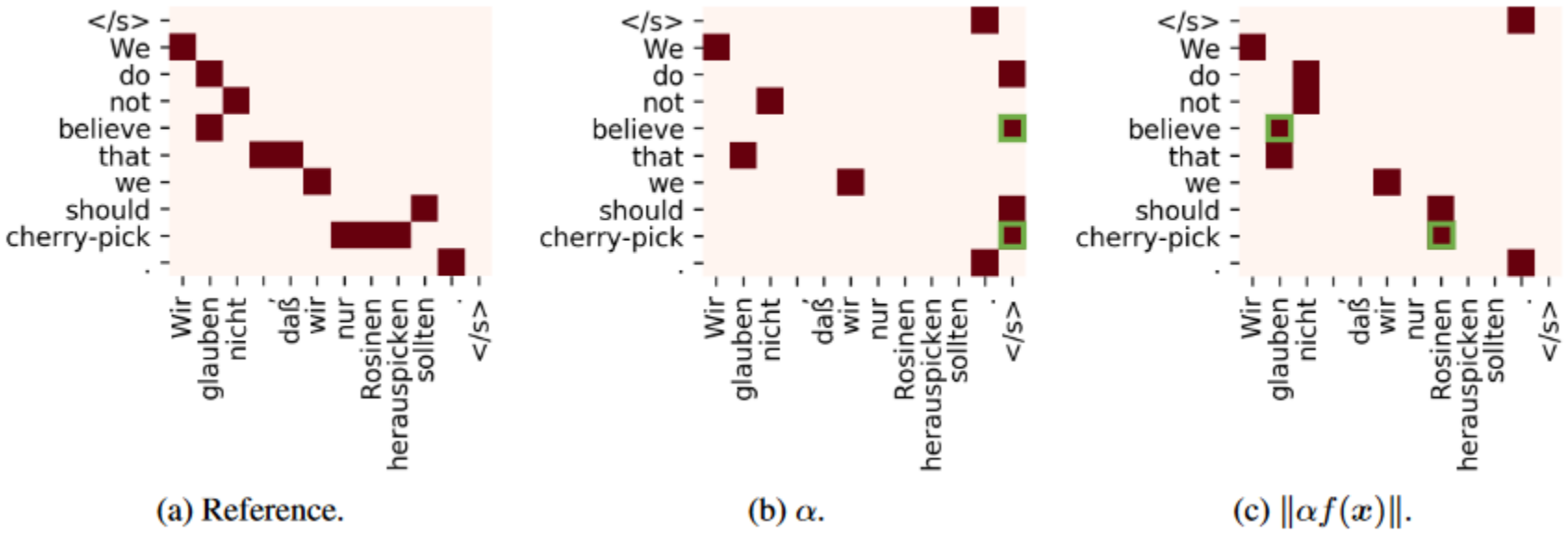
$$\underset{s_j}{\operatorname{argmax}}\, \alpha_{i,j} \qquad \underset{s_j}{\operatorname{argmax}}\, \|\alpha_{i,j} f(\boldsymbol{x}_j)\|.$$

- What about the alignment **proposed by a layer?** Integrating H heads within the same layer via summation

$$\underset{s_j}{\operatorname{argmax}} \sum_{h=1}^{H} \alpha_{i,j}^{h} \quad \text{or} \quad \underset{s_j}{\operatorname{argmax}} \| \sum_{h=1}^{H} \alpha_{i,j}^{h} f^{h}(\boldsymbol{x}_j) \|,$$

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Word Alignment for Neural Machine Translation

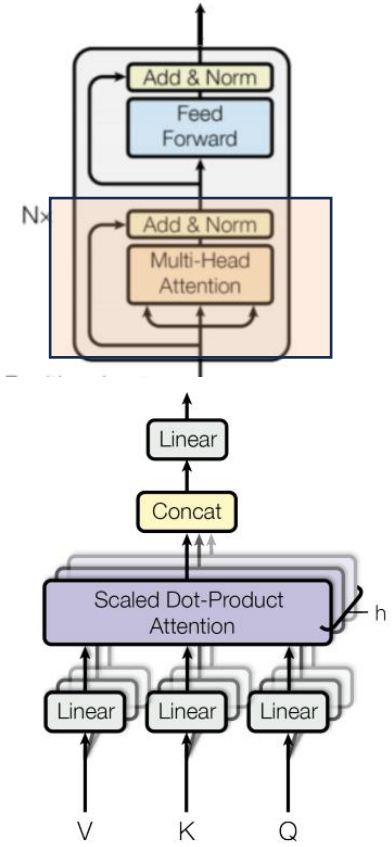


(a) Reference. (b) α. (c) $\|\alpha f(x)\|$.

Examples of the reference and extracted alignments using each method in layer. Misalignments are fixed

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Can we analyze the entire Attention Block?

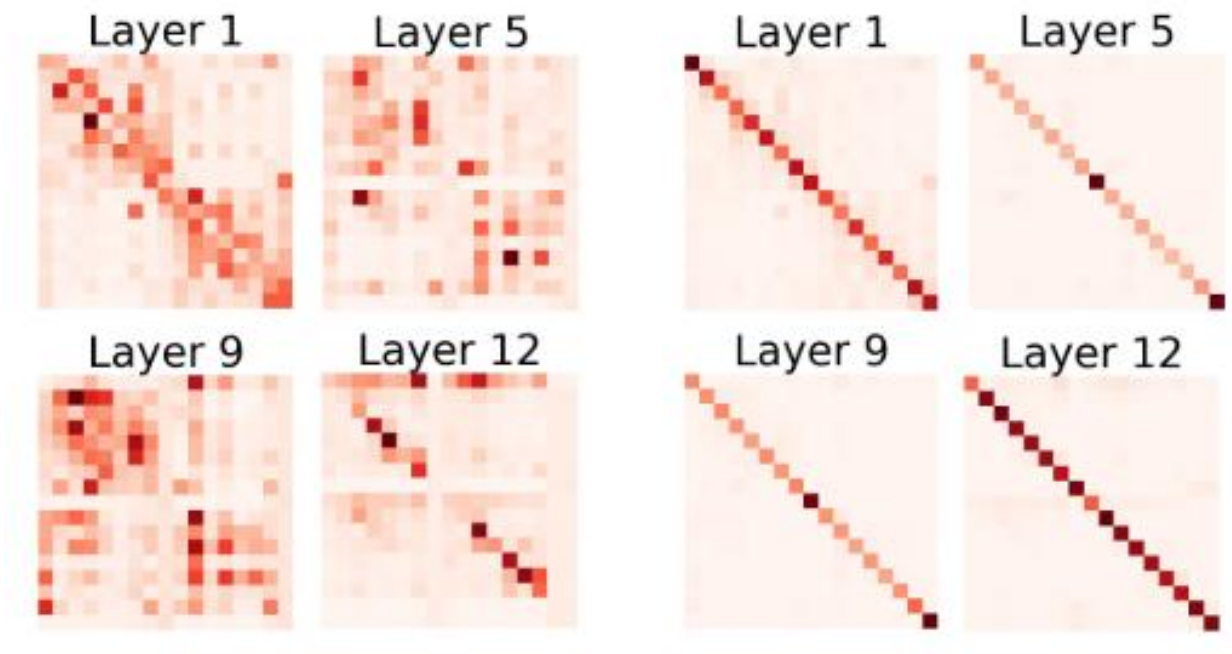


- The idea is to scale the study from the attention patterns to the whole attention block:
  - multi-head attention, residual connection, and layer normalization.
- Two contrasting effects of the attention block:
  - "**mixing**" the input representations via **attention**
  - "**preserving**" the original input mainly via **residual connection**
- Main results: smaller impact than expected for multi-head attention for "mixing"

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Can we analyze the entire Attention Block?

- The proposed ATTNRES-N and ATTNRESLN-N largely differ from those obtained from the existing methods
- The visualizations of the token-by-token interactions in the common attention map style become almost diagonal patterns



Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is Not Only a Weight: Analyzing Transformers with Vector Norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

The Feed Forward Module

Feed Forward layers as key-value memories

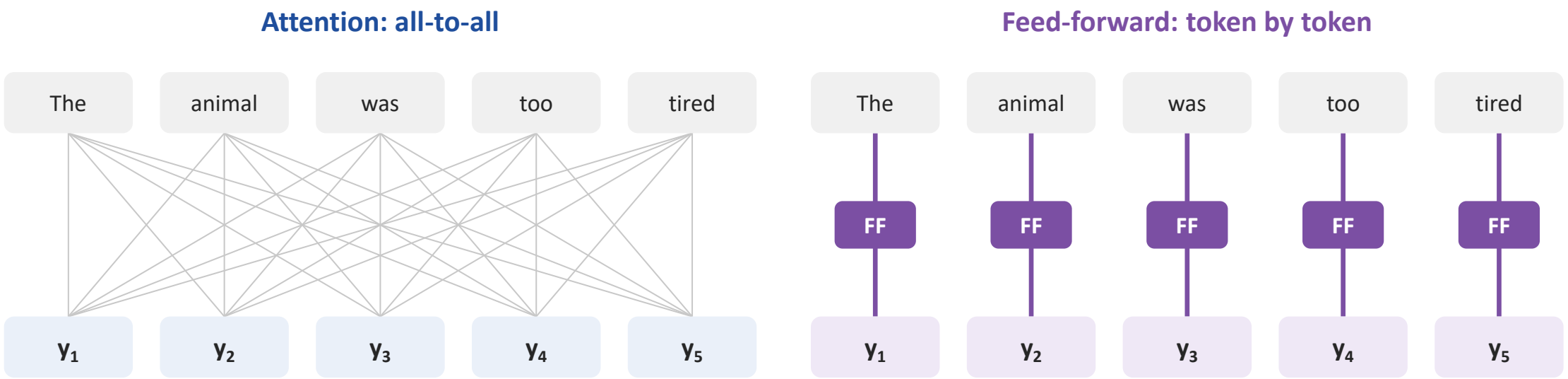
They account for >67% of the network parameters.

The Feed Forward Module

Feed Forward layers as key-value memories

They account for >67% of the network parameters.

Attention mixes tokens, the FF layer does not



The FF block is a position-wise function: the same two-layer network is applied to each token vector independently.

Information moves between tokens only through attention. The FF layer transforms what is already at each position.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

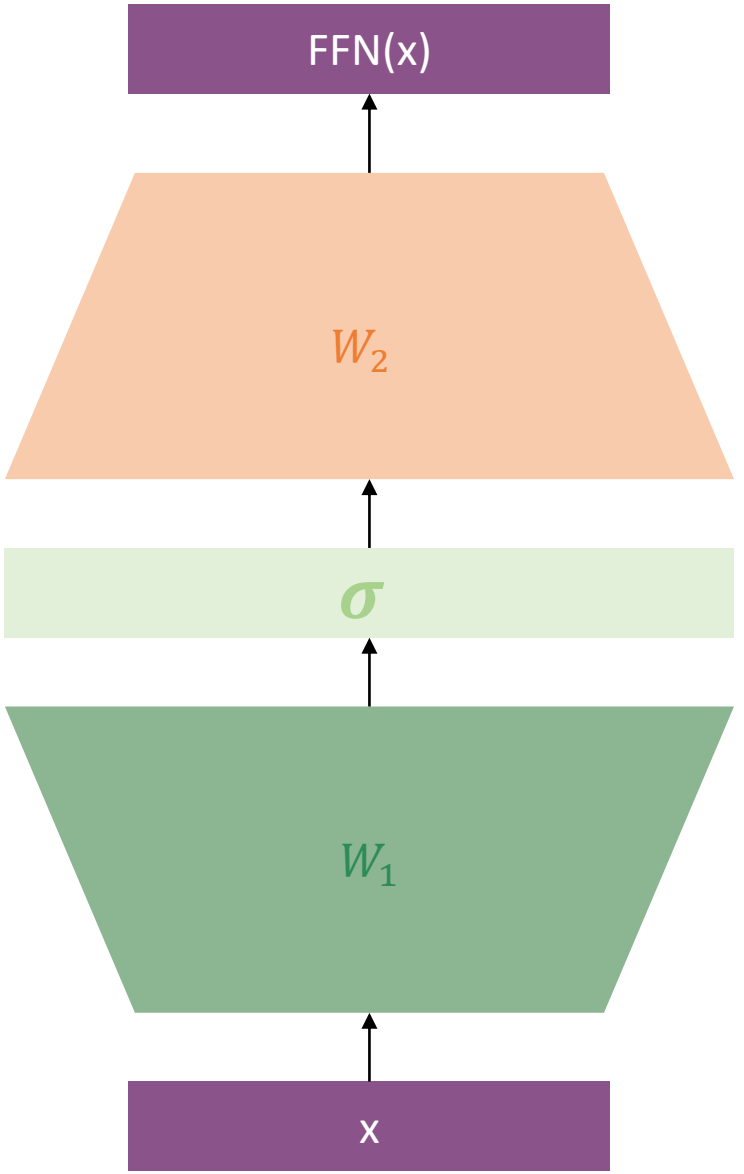
The FF block

For every token representation x

$$\mathrm{FFN}(x) = \sigma(xW_1{}^T)W_2$$

where:

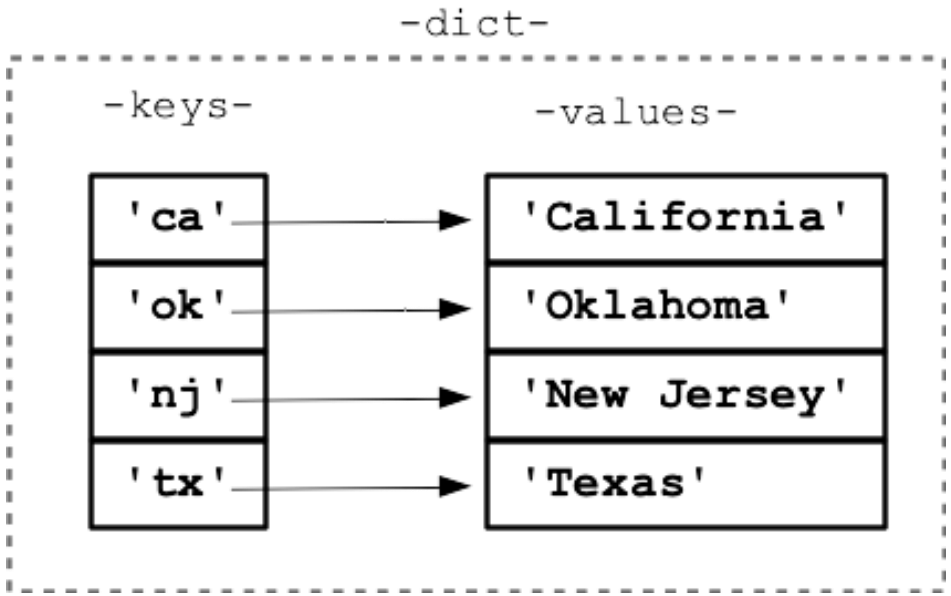
- W_1 expands the representation (typically 4× larger)
- σ = ReLU or GELU non-linearity
- W_2 projects back to the model dimension



Neural Memories

$$\text{FFN}(x) = \sigma(xW_1{}^T)W_2$$

- This equation reminds the one of **neural memories**!
- What are they?
  - Mappings (or dictionaries)
  - Associate a set of **keys** to the corresponding **values**



Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Neural Memories

- They store key-value pairs that we call **memories**
  - Keys $K \in \mathbb{R}^{d_m \times d}$
  - Values $V \in \mathbb{R}^{d_m \times d}$
  - Given an input in $\mathbb{R}^d$
    - Compute the similarities with the stored keys
    - Then retrieve the corresponding values

$$p(k_i \mid x) \propto \exp(\mathbf{x} \cdot \mathbf{k}_i)$$

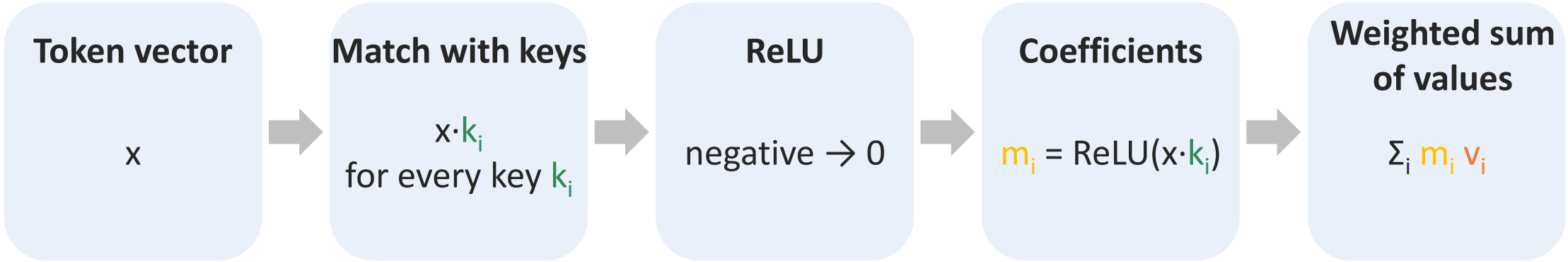
$$\mathbf{MN}(\mathbf{x}) = \sum_{i=1}^{d_m} p(k_i \mid x)\mathbf{v}_i$$

- In matrix form:

$$\mathbf{MN}(\mathbf{x}) = \text{softmax}(\mathbf{x} \cdot K^\top) \cdot V \quad \sim \quad \text{FFN}(x) = \sigma(xW_1{}^T)W_2$$

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

FF blocks emulate key-value memories



Same letters, different objects

q_i, k_j, v_j in attention: one per token j, computed from the input.

k_i, v_i in the FF layer: rows of two learned matrices, one per memory i, fixed after training.

$FF(x) = \text{ReLU}(x \cdot K^T) \cdot V$, where K and V are $d_m \times d$ matrices (biases omitted).

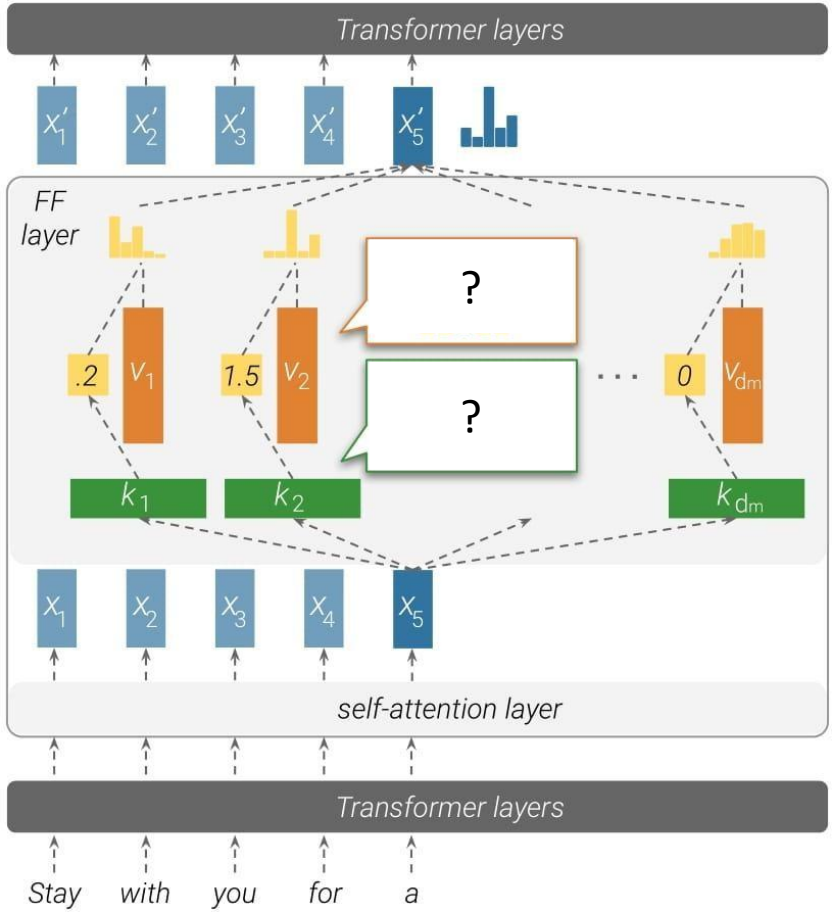
Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

FF blocks emulate key-value memories

- Each feed-forward layer is a position-wise function

$$\mathrm{FF}(\mathbf{x}) = f(\mathbf{x} \cdot K^\top) \cdot V$$

- That we suppose is storing key – value mappings
- **What is stored in those memories?**

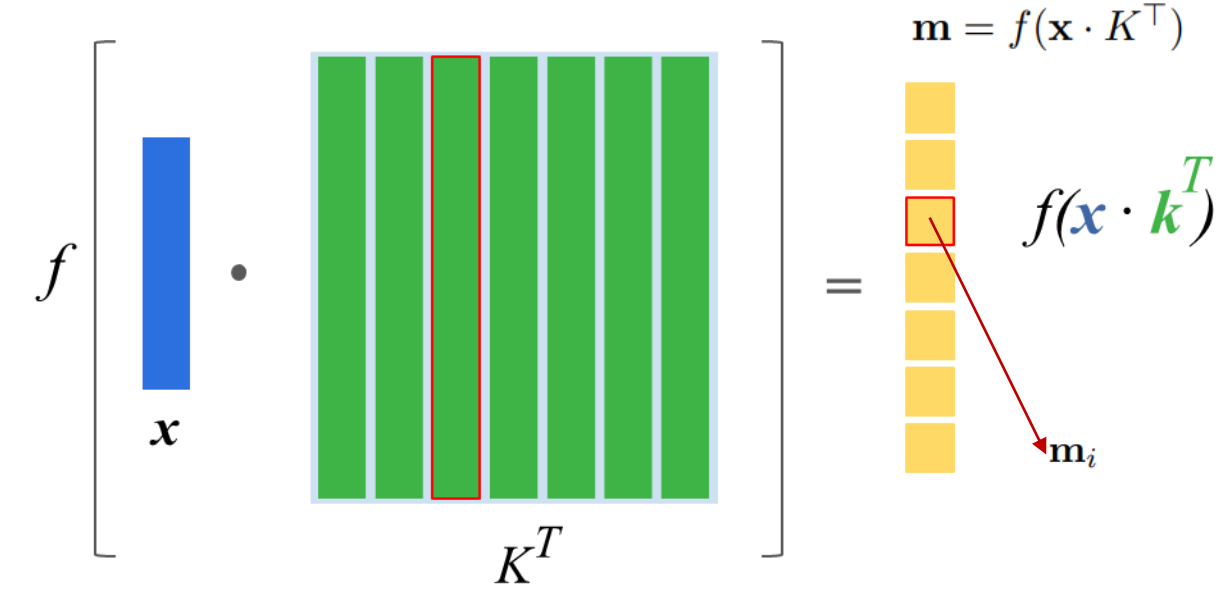


Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Given a **key**, look for trigger examples

Trigger examples are inputs that **activate** a key

- Define memory coefficients $\mathbf{m}_i$
- Compute memory coefficients for prefixes from WikiText
  - >> $\mathbf{m}_i$ → >> association
- Select top 25 triggers, and annotate



Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Layer 13, dimension 2989

@-@ central Kentucky, where several deadly tornadoes were reported during the evening hours between 7:30pm and 11

... often having completed several hours of work before arriving at the office between 9:30 and 10

A 1900 timetable shows that trams left for Manchester every 10 minutes between 8:00 am and 10

Beaches are open from Memorial Day to Labor Day with swimming permitted when lifeguards are on duty from 10

Polling stations are open from 7

The Gainesville Tornado of 1936 , one of the deadliest tornadoes in history , occurred at 8:30am local

Worldwide, most tornadoes occur in the late afternoon, between 3pm and 7

Weekend tolls are in effect from 7:00pm Friday until

The park is open daily from 7

... implemented additional airspace restrictions over Washington , D.C. between 10:00am and 6

Strengthening into a Category 4 system early on 27 March , Fay made landfall on the Pilbara coast between 8am and 9

In the hillside community of Mameyes , within Ponce , there was a block slide at around 3:00am local

He gave his resignation to the Congress at 19:45 local

This resulted in a red flag period, from 10:33am until 10

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Keys are activated by human-interpretable patterns

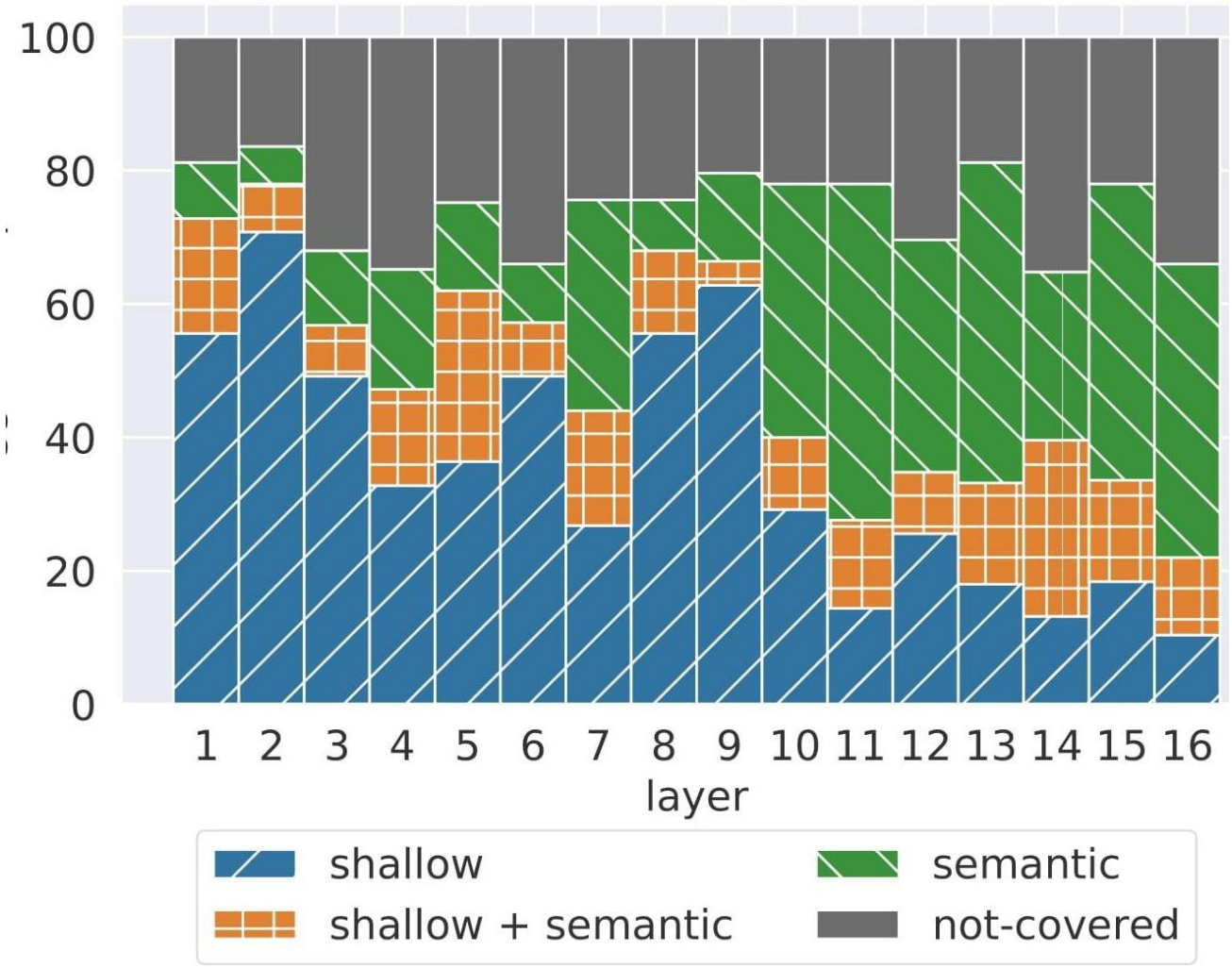
Key	Pattern	Trigger examples
layer 1, dim. 449	Ends with "substitutes" (shallow)	*At the meeting, Elton said that "for artistic reasons there could be no **substitutes*** *In German service, they were used as **substitutes*** *Two weeks later, he came off the **substitutes***
layer 6, dim. 2546	Military, ends with "base"/"bases" (shallow + semantic)	*On 1 April the SRSG authorised the SADF to leave their **bases*** *Aircraft from all four carriers attacked the Australian **base*** *Bombers flying missions to Rabaul and other Japanese **bases***
layer 10, dim. 2997	A "part of" relation (semantic)	*In June 2012 she was named **as one of** the team that competed* *He was also **a part of** the Indian delegation* *Toy Story is also **among the top ten** in the BFI list of the 50 films you should*

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Most trigger examples are associated with a pattern

Lower layers detect shallow patterns

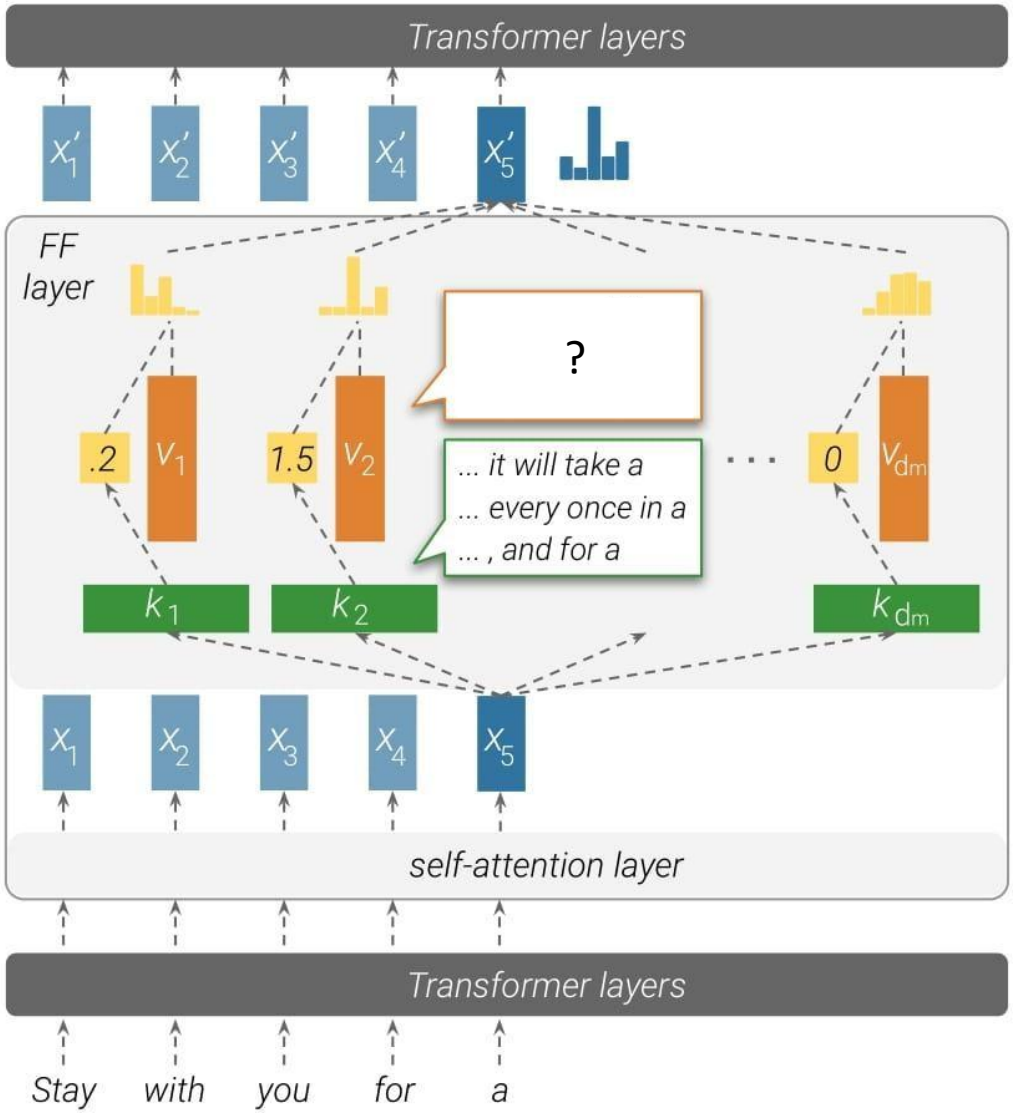
Higher layers (after layer 10) detect semantic patterns



Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

So **keys** capture input patterns

What do **values** represent?



Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Values induce a distribution over output vocabulary

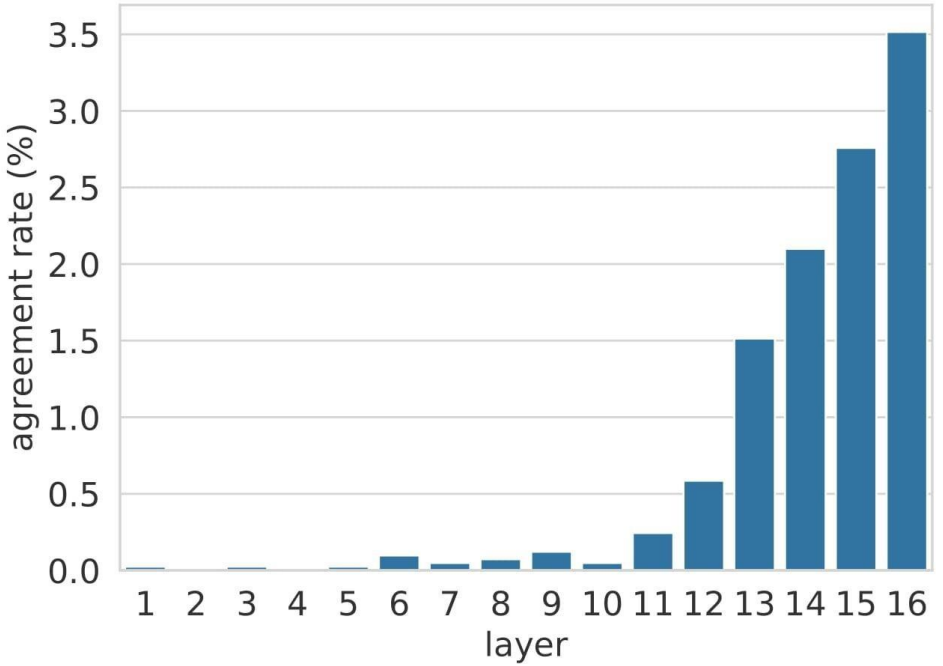
Compare the **value** prediction with the next token of the top trigger example

Make a vocabulary projection

He was also a part of the Indian | *delegation*

top trigger example of ***k*** *its next token*

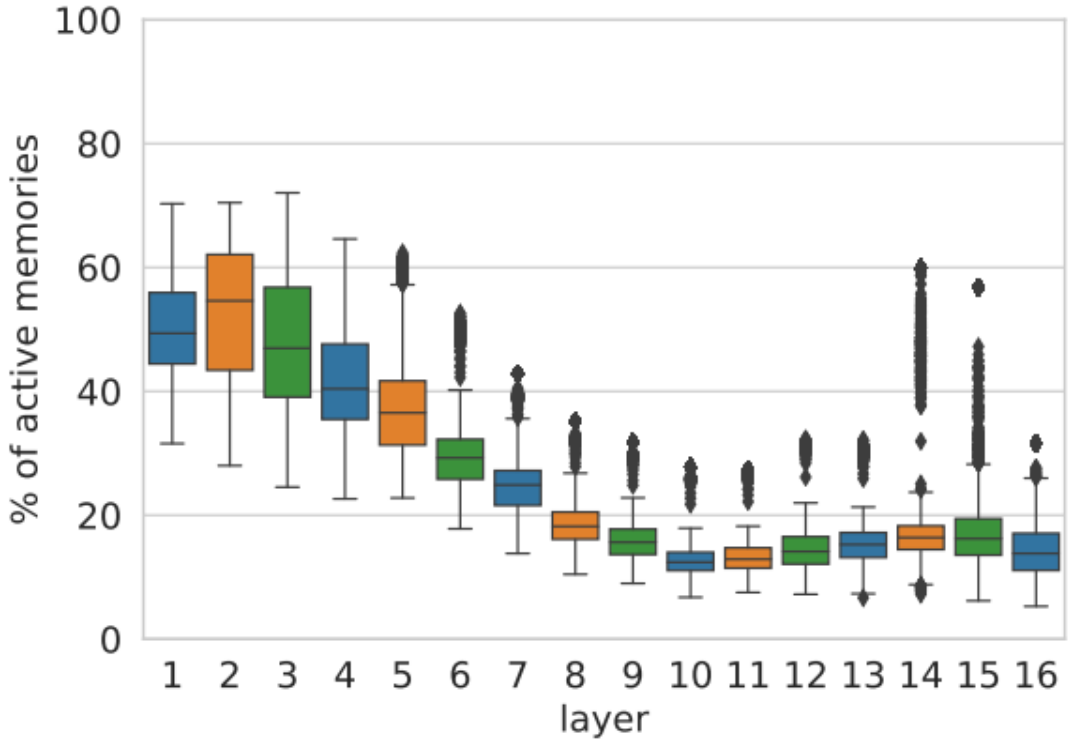
$$\mathbf{p}_i^\ell = \text{softmax}(\mathbf{v}_i^\ell \cdot E)$$



A random guess will have far lower agreement rate (0.0004%)

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

How does the model compose the output from **multiple memories in a layer**?

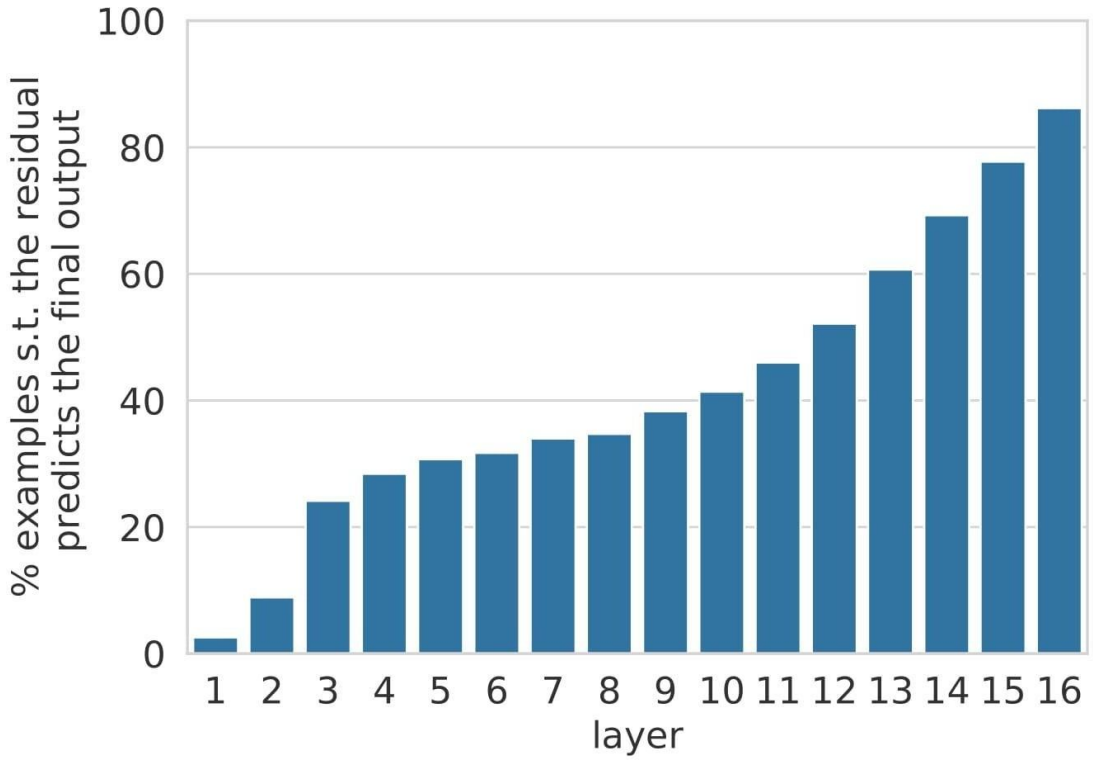


- Measure of active memories
- 10-50% of the memories are active
- Active memories drop after layer 10
  - Semantics?

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Building final prediction requires to include the residual stream

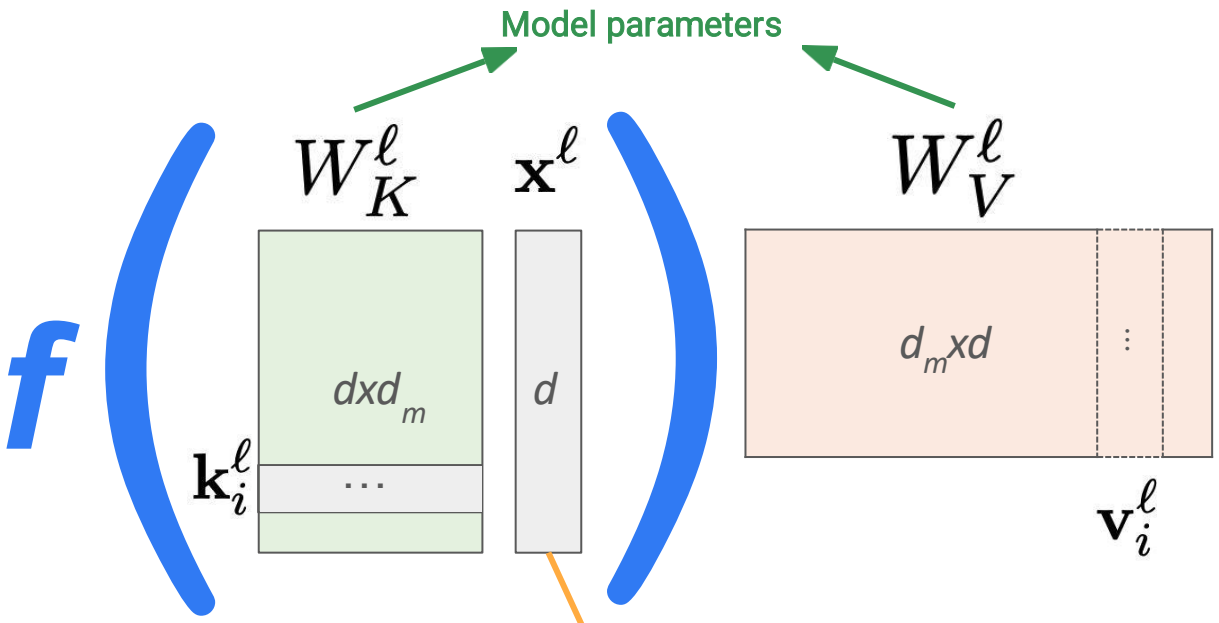
- The output of each layer is the sum of the FF layer's output and the residual
- ***output = residual + ffn***
- At which layer does the residual contain the model's final prediction?



Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

What is the mechanism that controls for the update?

- Focus on the operation at each layer
- Decompose the output of the layer, is a weighted sum



Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Layerwise Distribution Over the Vocabulary

Layerwise next token distribution

$$\tilde{\mathbf{p}}_i^\ell = \text{softmax}(E\mathbf{x}_i^\ell + E\mathbf{o}_i^\ell)$$
$$= \text{softmax}(E\mathbf{x}_i^\ell + E\sum_{i=1}^{d_m} m_i^\ell \mathbf{v}_i^\ell)$$

What is the effect of a single sub-update?

$$p(w \mid \mathbf{x}^\ell + m_i^\ell \mathbf{v}_i^\ell, E) = \frac{\exp\left(\mathbf{e}_w \cdot \mathbf{x}^\ell + \mathbf{e}_w \cdot m_i^\ell \mathbf{v}_i^\ell\right)}{Z\left(E(\mathbf{x}^\ell + m_i^\ell \mathbf{v}_i^\ell)\right)}$$
$$\propto p(w \mid \mathbf{x}^\ell, E) \cdot \exp\left(\mathbf{e}_w \cdot m_i^\ell \mathbf{v}_i^\ell\right)$$

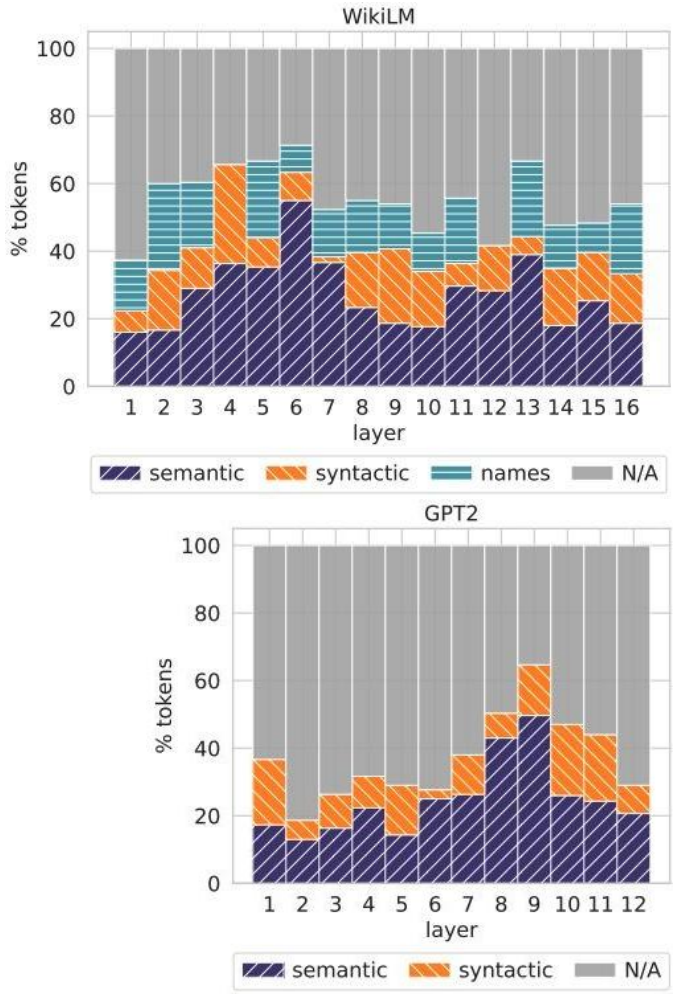
Value vectors are responsible for scaling the residual probability of every token

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

FFNs values encode concepts in the vocabulary space

- Sample many value vectors
- Project them into the embedding matrix
- Obtain top-30 scoring tokens
- Identify patterns that occur in at least 4 tokens
- Classify as either “semantic” or “syntactic”

Value	Top-scoring tokens	Pattern
GPT2 layer 3, dim. 1018	kg, percent, spread, total, yards, pounds, hours, ms, overall, kilograms, hectares, IU, cumulative	Measurement (semantic)
GPT2 layer 8, dim. 1900	which, whose, Which, whom, where, who, wherein, whence, whereas, Whereas	WH-relativizers (syntactic)
WikiLM layer 6, dim. 3025	largely, rapidly, effectively, previously, normally, quickly, easily, potentially	Adverbs (syntactic)
WikiLM layer 13, dim. 3516	policymakers, geneticists, ancestries, Ohioans, Oklahomans, herpetologists, hydrologists	Groups of people (semantic)



Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

FFN values influence the output distribution

FFN Values Encode Concepts that Promote Tokens

$$p\left(w \mid \mathbf{x}^\ell + m_i^\ell \mathbf{v}_i^\ell, E\right) = \frac{\exp\left(\mathbf{e}_w \cdot \mathbf{x}^\ell + \mathbf{e}_w \cdot m_i^\ell \mathbf{v}_i^\ell\right)}{Z\left(E(\mathbf{x}^\ell + m_i^\ell \mathbf{v}_i^\ell)\right)}$$

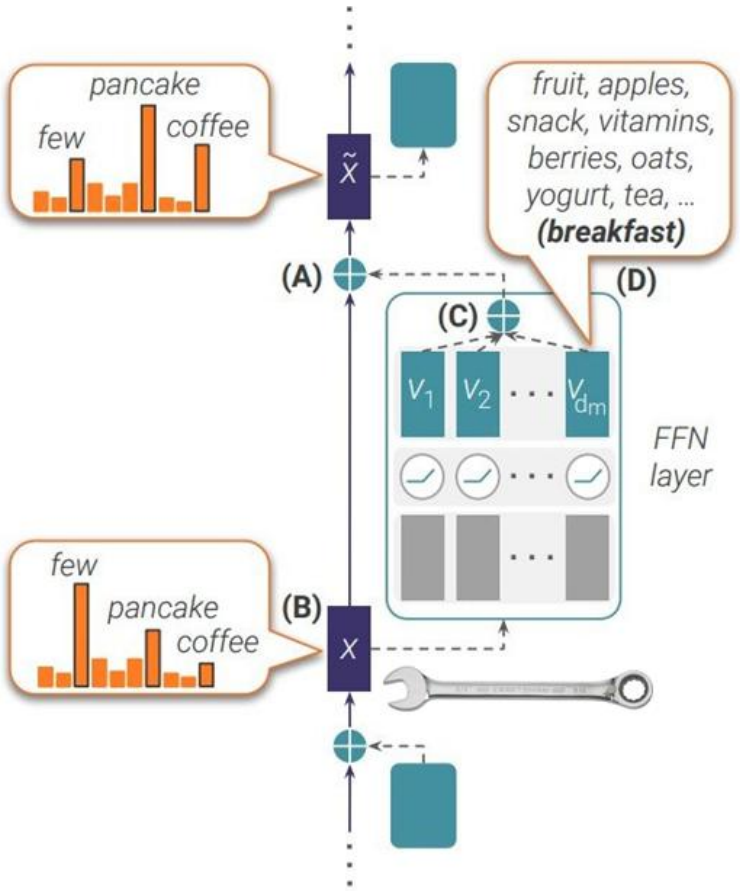
$$\propto p\left(w \mid \mathbf{x}^\ell, E\right) \cdot \exp\left(\mathbf{e}_w \cdot m_i^\ell \mathbf{v}_i^\ell\right)$$

$$w \uparrow \quad \mathbf{e}_w \cdot m_i^\ell \mathbf{v}_i^\ell > 0$$

$$w \downarrow \quad \mathbf{e}_w \cdot m_i^\ell \mathbf{v}_i^\ell < 0$$

Values and coefficients influence specific tokens

This score influence scale the probability of one word



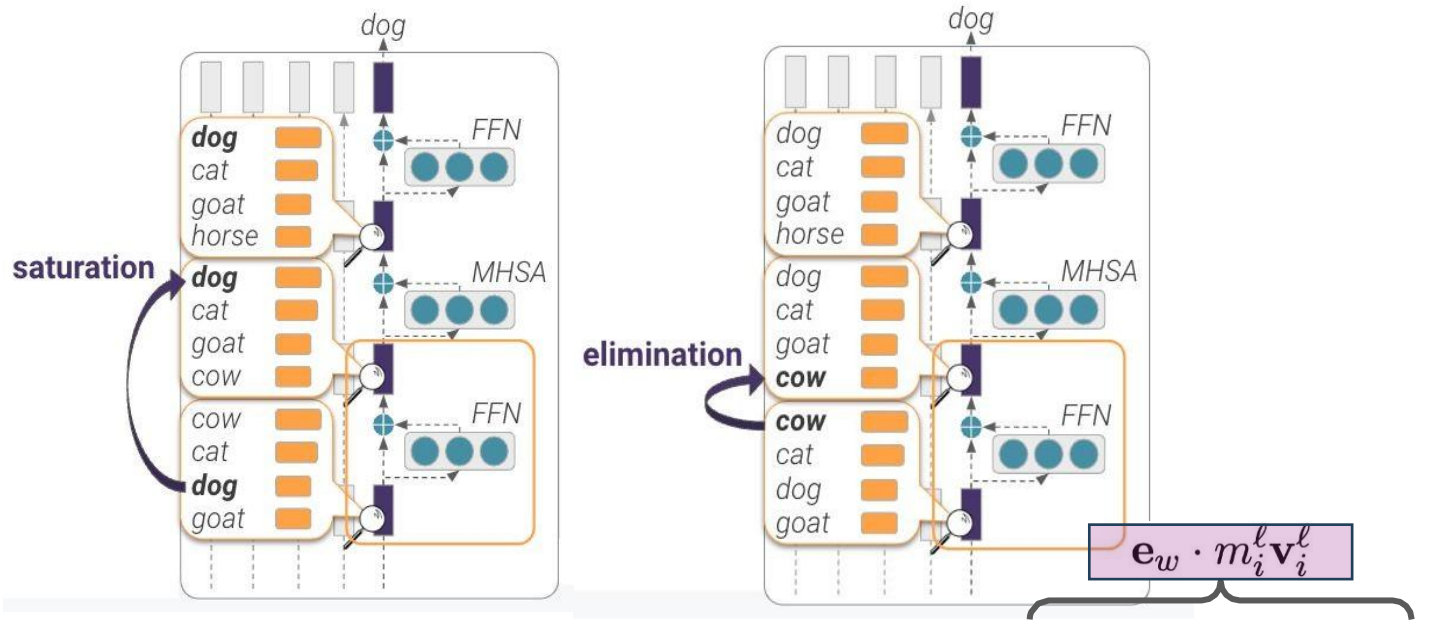
Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

FFN values influence the output distribution

- Consider the 10 most **dominant sub-updates** in each layer
  - that is, the 10 sub-updates the largest contribution to the representation

$$|m_i^\ell| \cdot ||\mathbf{v}_i^\ell||$$

- What are the scores for the reference token?
- Tokens are pushed strongly by a few dominant values
- FFN updates work in a promotion mechanism:
  - In both models, tokens promoted to the top of the distribution receive higher maximum scores than tokens eliminated from the top position



Sub-updates	Event	Max.	Mean	Min.
WIKILM, dominant	*saturation*	1.2	< 0.01	−0.8
	elimination	0.5	−0.01	−0.5
GPT2, dominant	*saturation*	8.5	1.3	−4.9
	elimination	4.0	0.1	−3.6

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Takeaways

We discussed that FF layers emulate **key**-**value** memories

keys capture textual pattern(s) in the input

values represent distributions over the output vocabulary

The output of each FF layer is a **composition** of multiple **values**

The outputs of the FF layers **refine** the model's prediction across layers

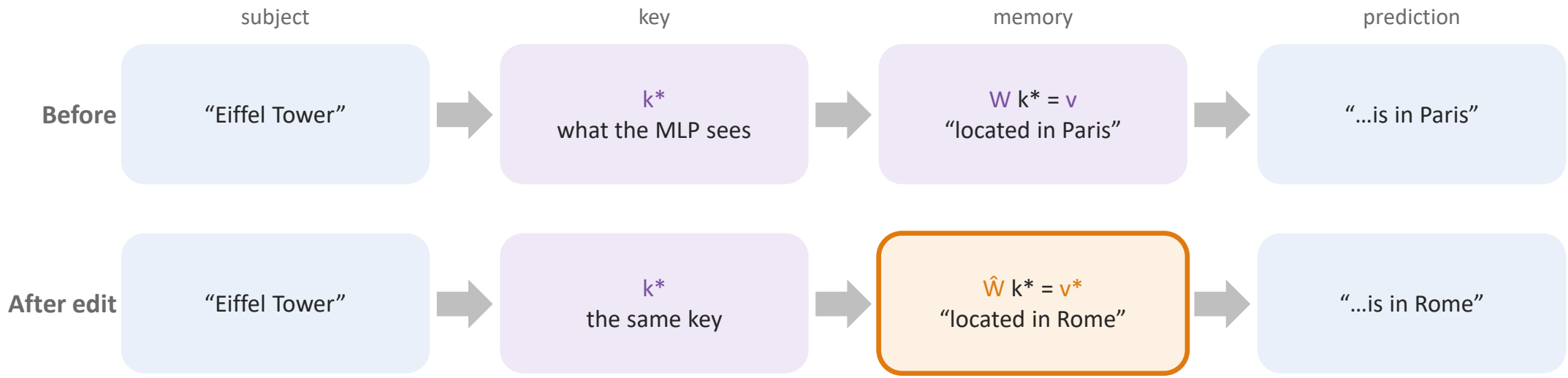




Editing Memories

How to modify the memory of a Transformer

Editing = rewriting one key–value pair



Editing a fact means changing what the memory returns for one key, and leaving every other key alone.

Two questions: where is the fact stored (locate), and how do we change W for that key only (edit)?

Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

ROME in three steps

1 Locate

Where is the fact stored?

Causal tracing finds the mid-layer MLP, at the last subject token, that carries the fact.

2 Pick the value

What should the memory return?

Optimise a vector v^* so that, when it replaces the MLP output there, the model says “Rome”.

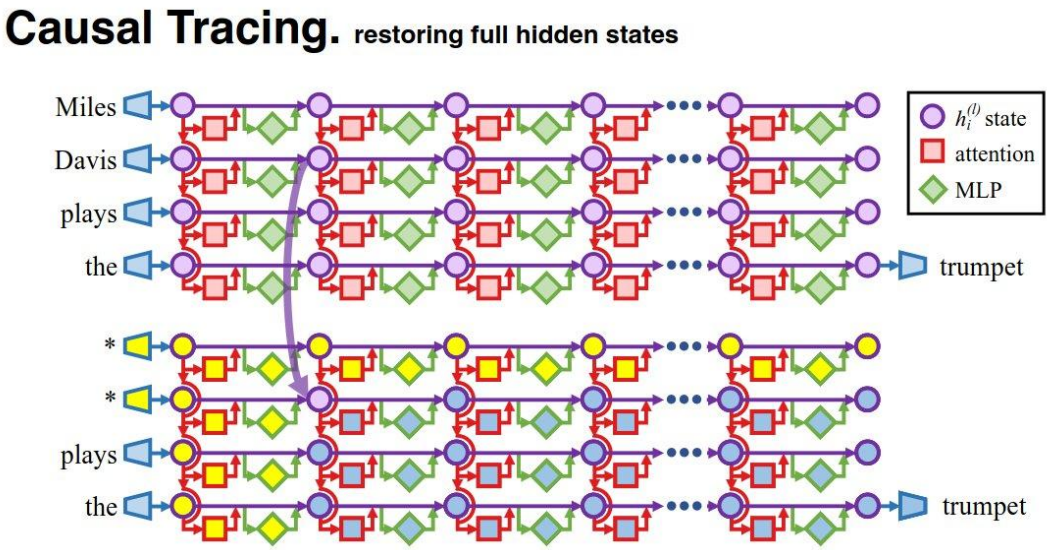
3 Rewrite the weights

How do we change W?

Change W minimally so that $W k^* = v^*$, using a rank-one update that leaves other keys almost untouched.

Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

Locate - Where is the fact stored?



- Is there a way to find a component responsible for a generation?
- The principle is “locate first, then edit”
- View the computation of a Transformer-based model as computational graph
- Remove nodes that are important

Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

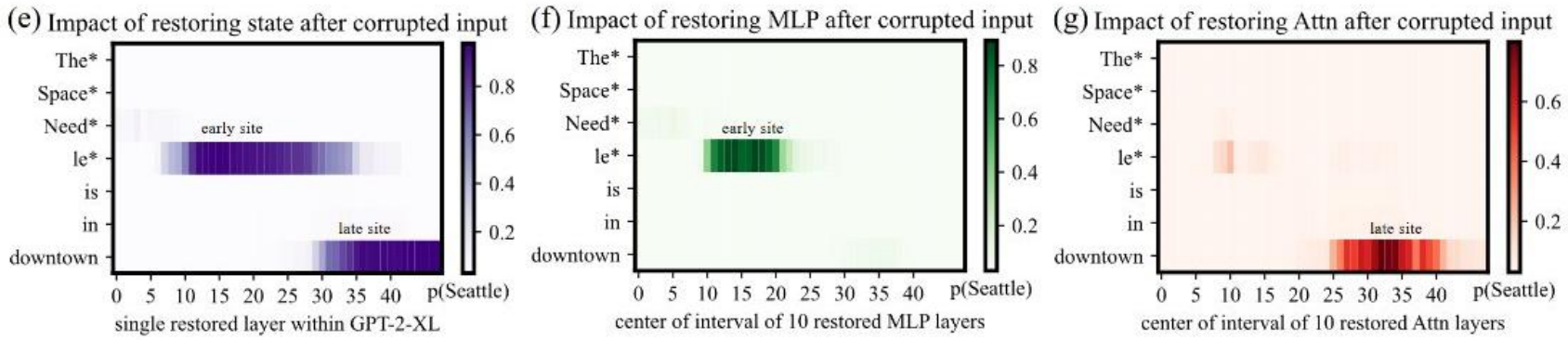
Locate - Where is the fact stored?

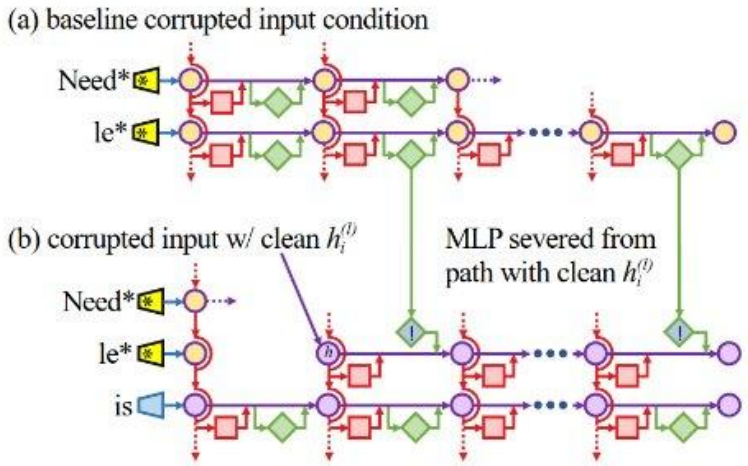


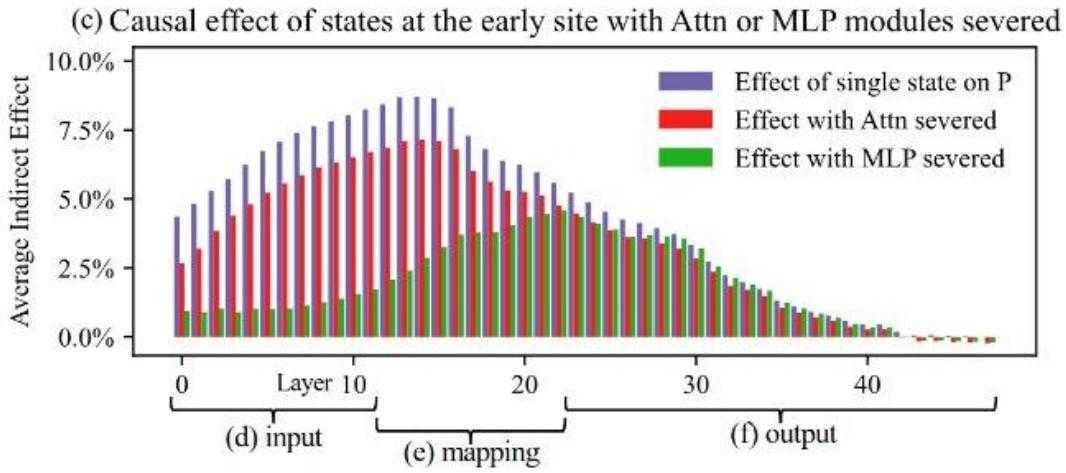
- A clean run that predicts the fact.
- A corrupted run where the prediction is damaged
- A corrupted-with-restoration run

Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

Locate – The Feed Forward blocks impact







Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

Edit – Closed form solution

- If it is true that a matrix is implementing a neural memory to maps keys to values, then the optimization problem to memorize the mapping is:

$$W_{out}^{l} = \arg\min_{\hat{W}} \left(\sum_{(k,v):k\in K_0, v\in V_0} \left\| \hat{W}k - v \right\|^2 \right)$$

- Can be solved in closed form!

$$\hat{W} = W_{out}^{l} + \Lambda (C_0^{-1} k^*)^T$$

$$C_0 = K_0 K_0^T = (K_0 K_0)^T$$

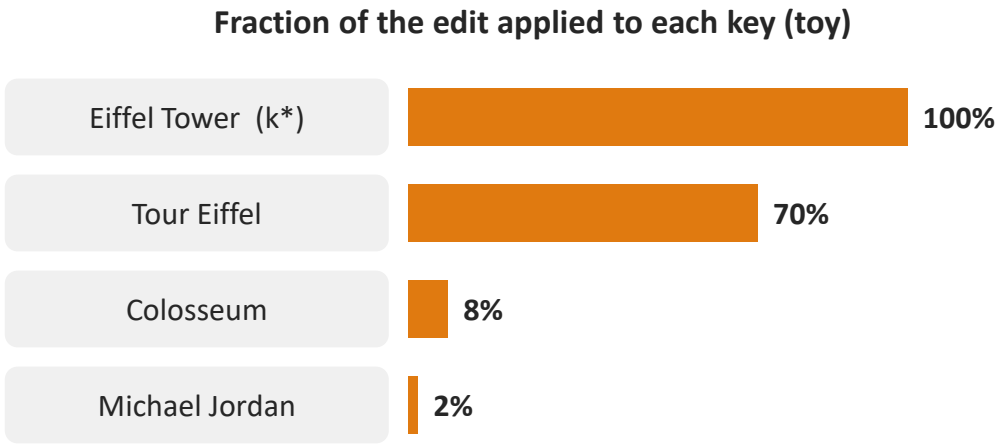
$$\Lambda = \frac{v^* - W_{out}^{l} k^*}{(C_0^{-1} k^*)^T k^*}$$

Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

Why rank-one edits are selective

$$\Delta W = \Lambda\, u^{T} \quad \text{with} \quad u = C^{-1}k^{*}$$

- For any key k, the output changes by $\Lambda \cdot (u \cdot k)$.
- Keys that look like k^{*} (large $u \cdot k$) are shifted a lot.
- Unrelated keys ($u \cdot k \approx 0$) are barely touched.
- $C = K_0 K_0^{T}$: statistics of the keys already stored, so C^{-1} steers the edit away from busy directions.



One matrix write, but a selective one: the edit follows the similarity to k*.

Illustrative values. Derivation in the backup slides Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

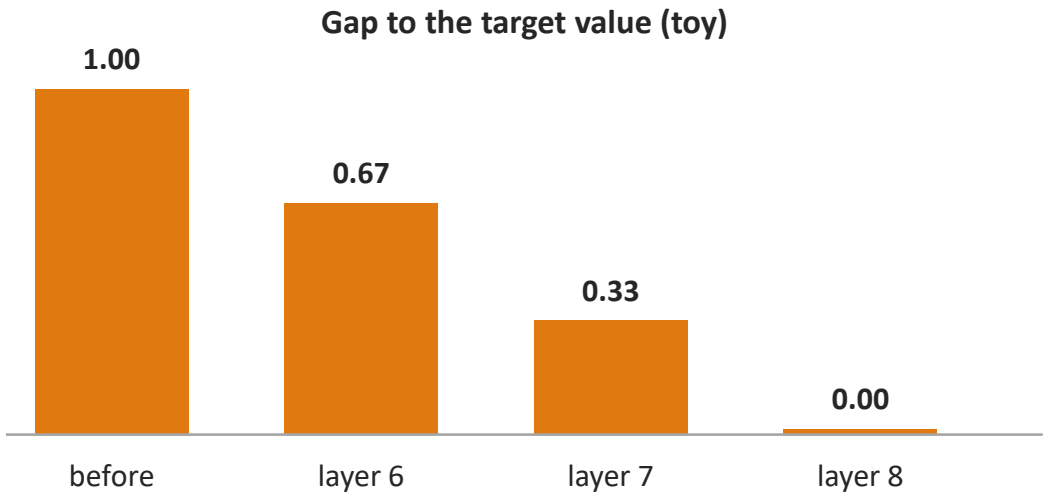
What if one wants to edit multiple memories?

ROME: one fact, one layer

Applied one by one, many edits start to interfere (see the plot).

MEMIT: n facts, several layers

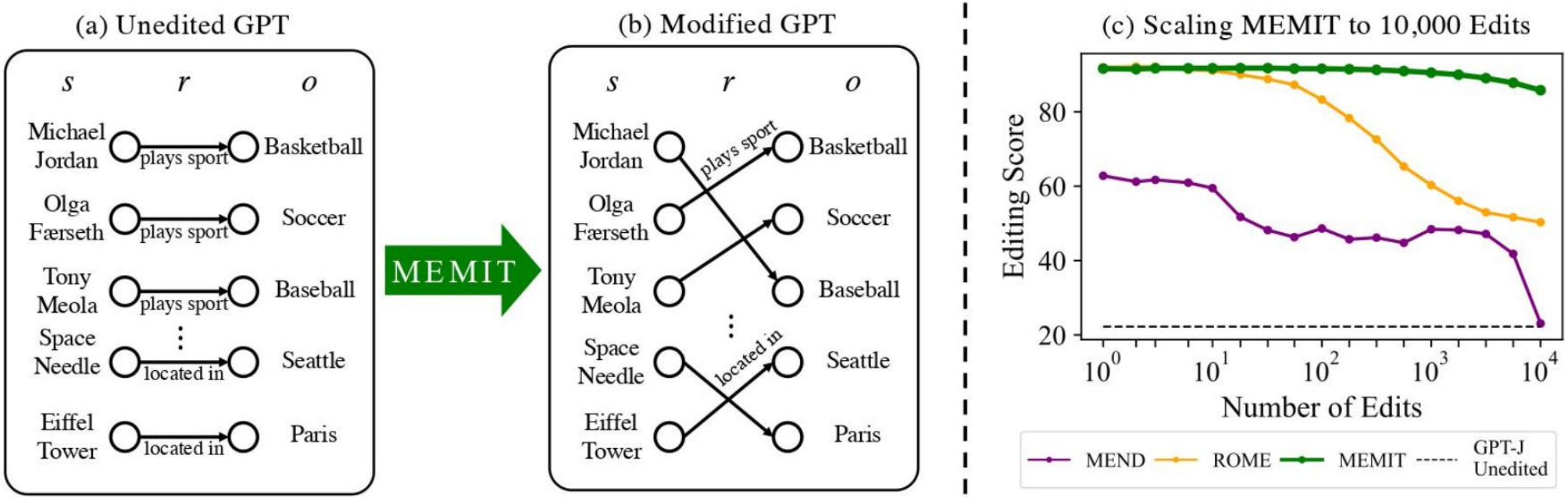
1 Find a target value for each fact
2 Split the needed change across layers
3 Solve one least-squares problem per layer, for all facts together



Each layer makes a smaller change, and all the facts are inserted in one batch.

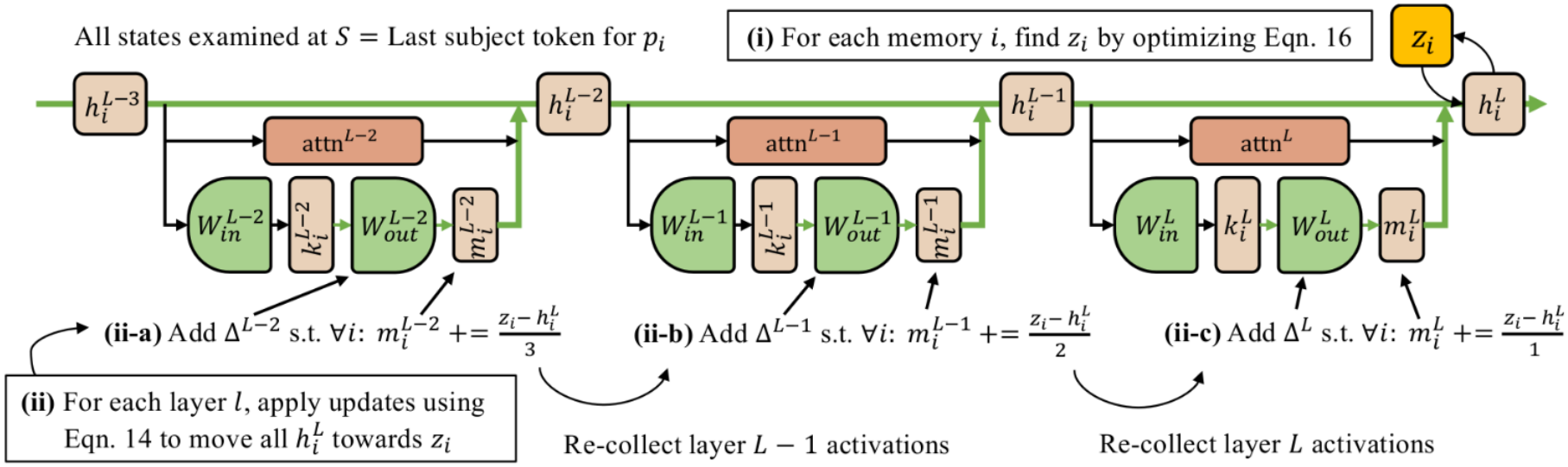
Illustrative example. Kevin Meng, David Bau, Alex Andonian, Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, *35*, 17359-17372.

What if one wants to edit multiple memories?



Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov and David Bau. 2023. Mass-Editing Memory in a Transformer. In *The Eleventh International Conference on Learning Representations*

What if one wants to edit multiple memories?



$$\Delta^l(K_0 K_0^T + K^* K^{*T}) = (V^* - W_{out}^l K^*) K^{*T}$$

Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov and David Bau. 2023. Mass-Editing Memory in a Transformer. In *The Eleventh International Conference on Learning Representations*





Editing Memories

How to modify the memory of a Transformer

The Role of the Attention Mechanism +

The Feed Forward Module +

Editing Memories =

Feed-forward seems to memorize (possibly generalized) patterns

Alternative Architectures for Language Models

Alternative Architectures for Language Models



How to build a Language Model

Transformers learn



Feed-forward seems to memorize (possibly generalized) patterns

How to build a Language Model

Paradigm shift

Feed-forward seems to memorize (possibly generalized) patterns



Transformers learn

How to build a Language Model

Paradigm shift

memorize (possibly generalized) patterns



Possibly learn

Correlation Matrix Memories (CMM)

+ Nearly orthogonal random vectors

Correlation Matrix Memories (CMM)

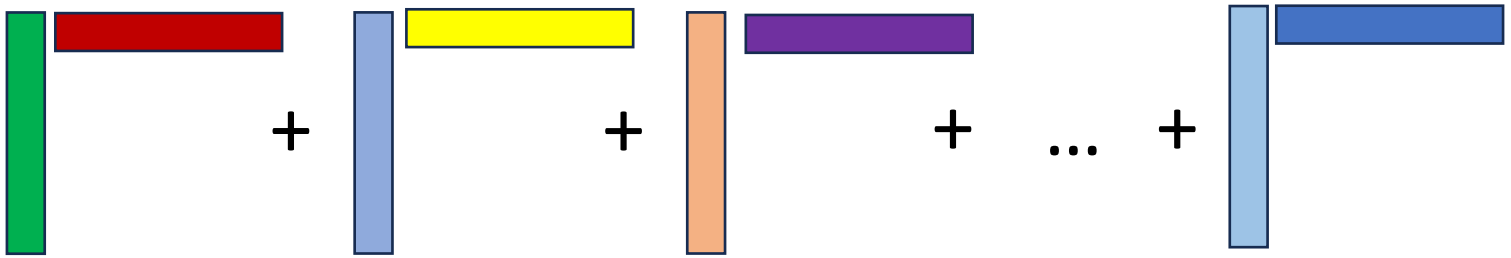
Memory = {(key,value)}

$$C = \sum_{i=1}^{n} \mathbf{k}_i \mathbf{v}_i^T$$

Correlation Matrix Memories (CMM)

Memory = {(key,value)}

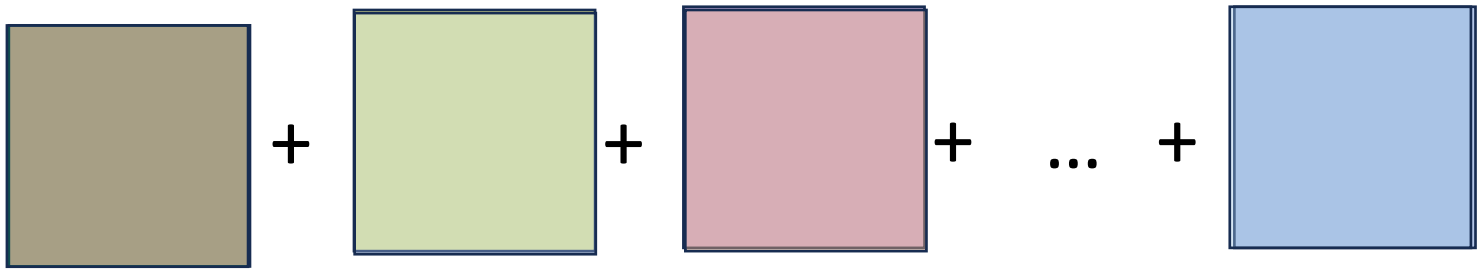
$$C = \sum_{i=1}^{n} \mathbf{k}_i \mathbf{v}_i^T$$

CMM =  + + + ... +

Correlation Matrix Memories (CMM)

Memory = {(key,value)}

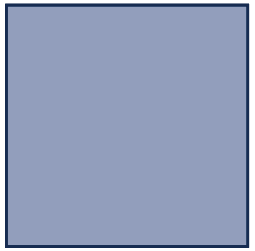
$$C = \sum_{i=1}^{n} \mathbf{k}_i \mathbf{v}_i^T$$

CMM =  + ... +

Correlation Matrix Memories (CMM)

Memory = {(key,value)}

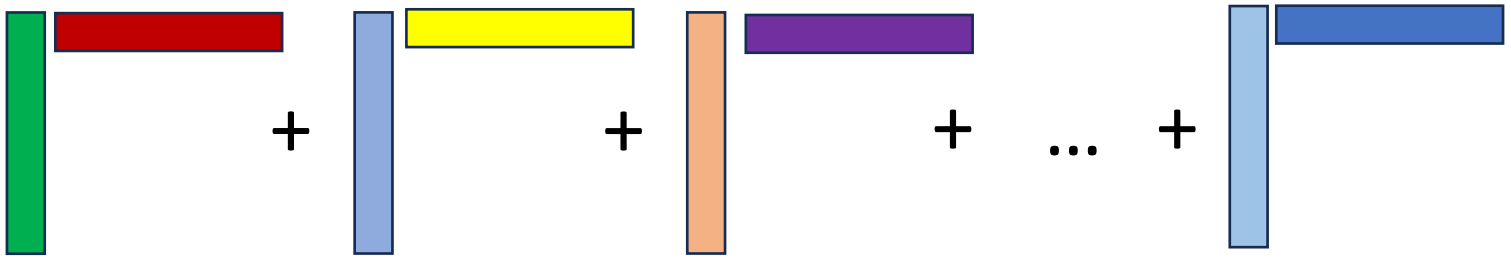
$$C = \sum_{i=1}^{n} \mathbf{k}_i \mathbf{v}_i^T$$

CMM = 

Correlation Matrix Memories (CMM)

Memory = {(key,value)}

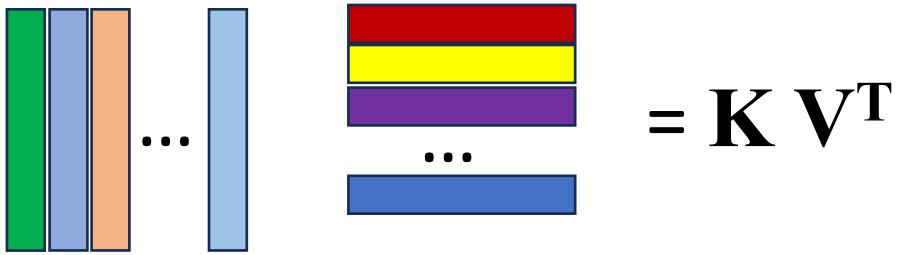
$$C = \sum_{i=1}^{n} \mathbf{k}_i \mathbf{v}_i^T$$

CMM =  + + + ... +

Correlation Matrix Memories (CMM)

Memory = {(key,value)}

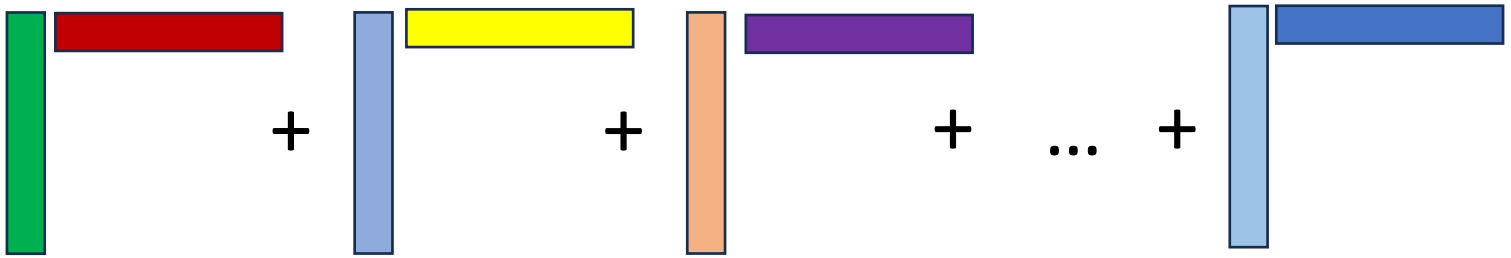
$$C = \sum_{i=1}^{n} \mathbf{k}_i \mathbf{v}_i^T$$

CMM =  = $\mathbf{K}\,\mathbf{V}^{\mathbf{T}}$

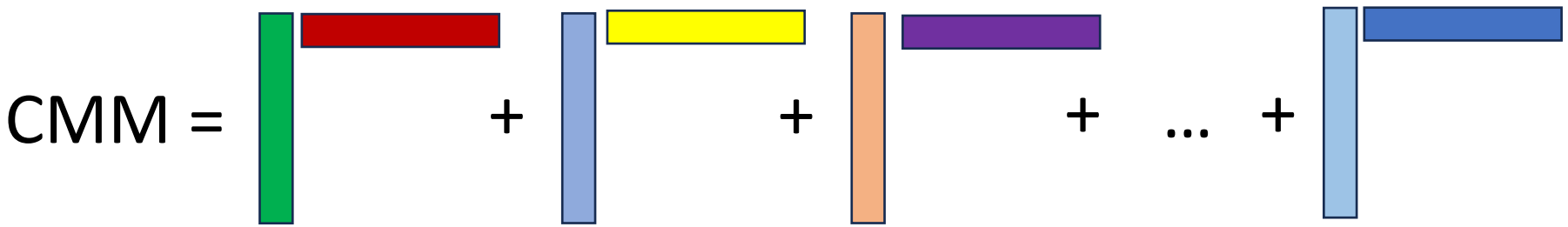
Correlation Matrix Memories (CMM)

Memory = {(key,value)}

$$C = \sum_{i=1}^{n} \mathbf{k}_i \mathbf{v}_i^T$$

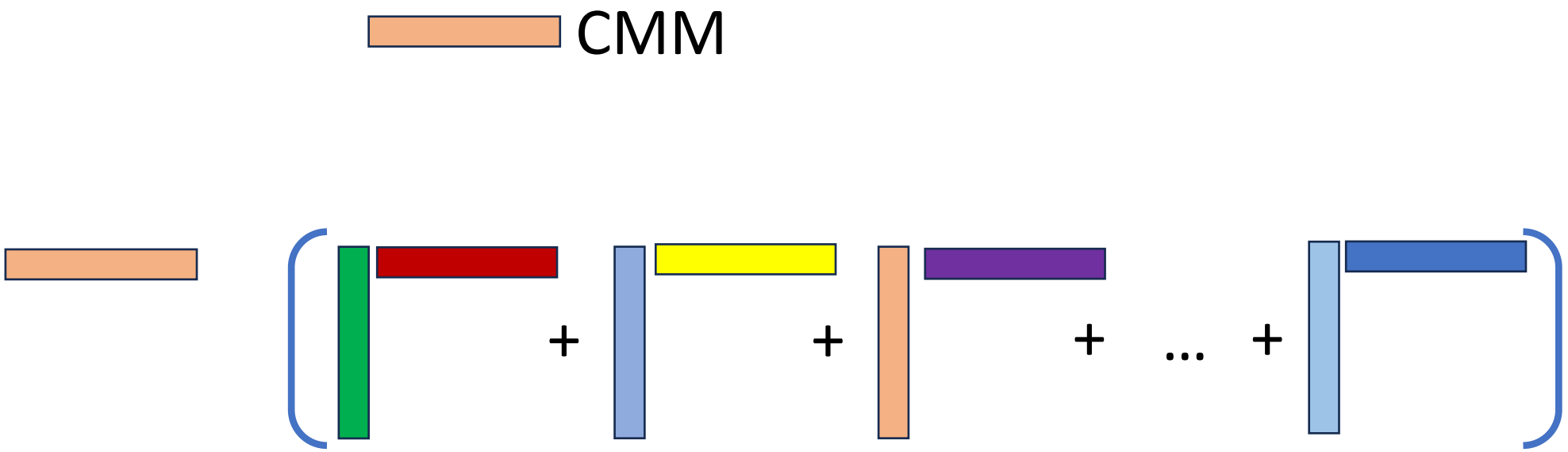
CMM =  + + + ... +

Correlation Matrix Memories (CMM)

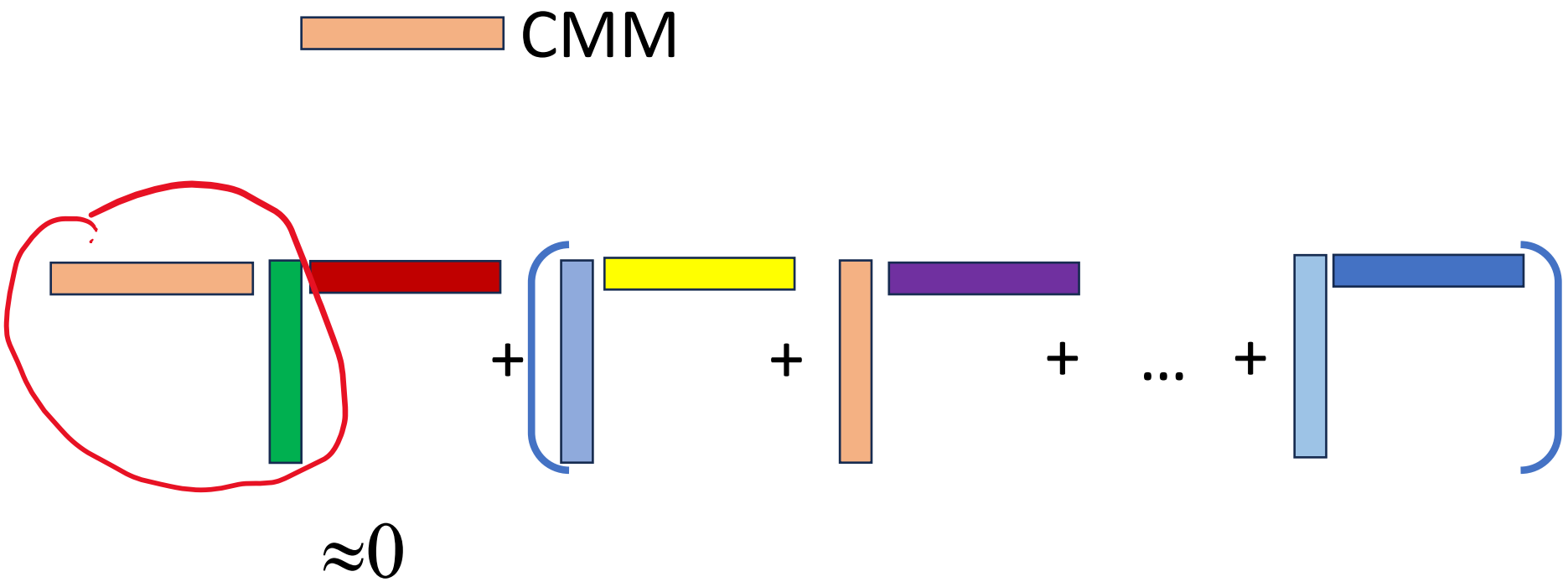




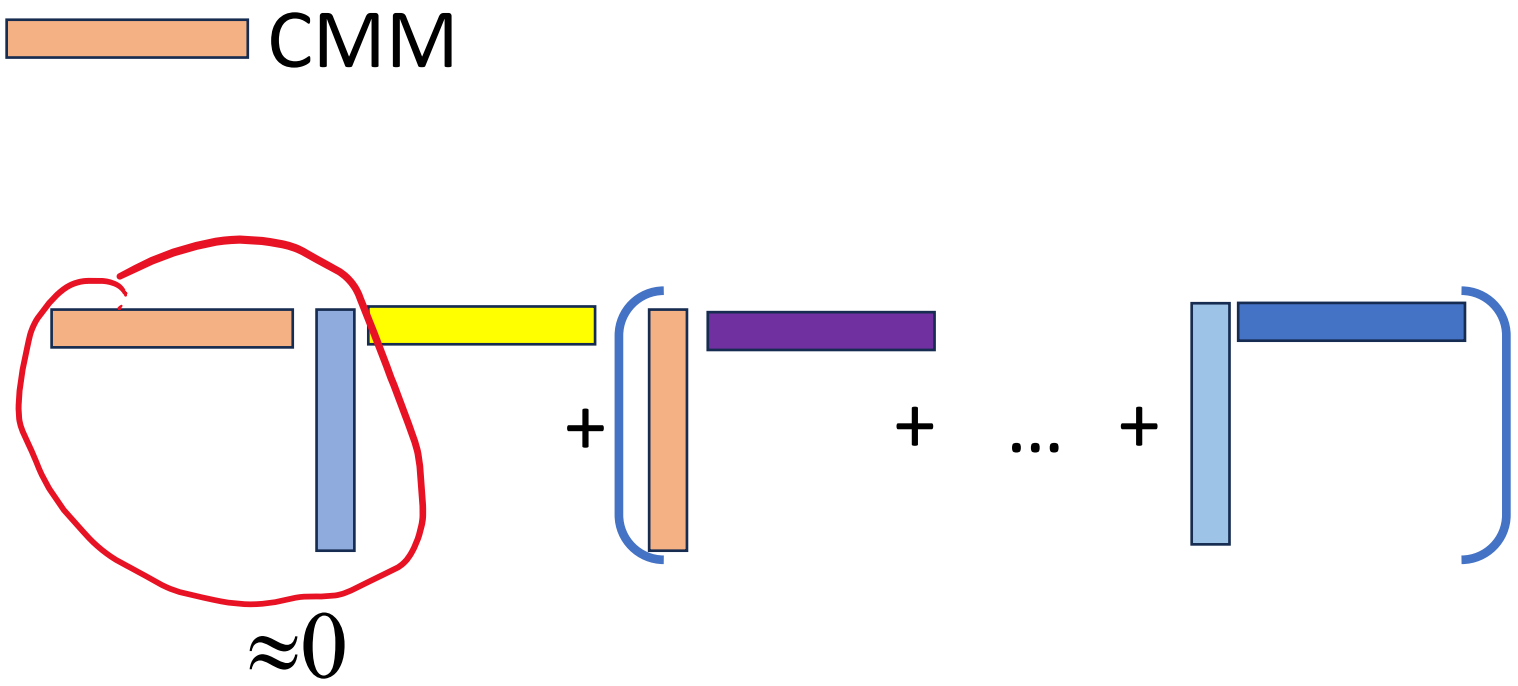
Correlation Matrix Memories (CMM)



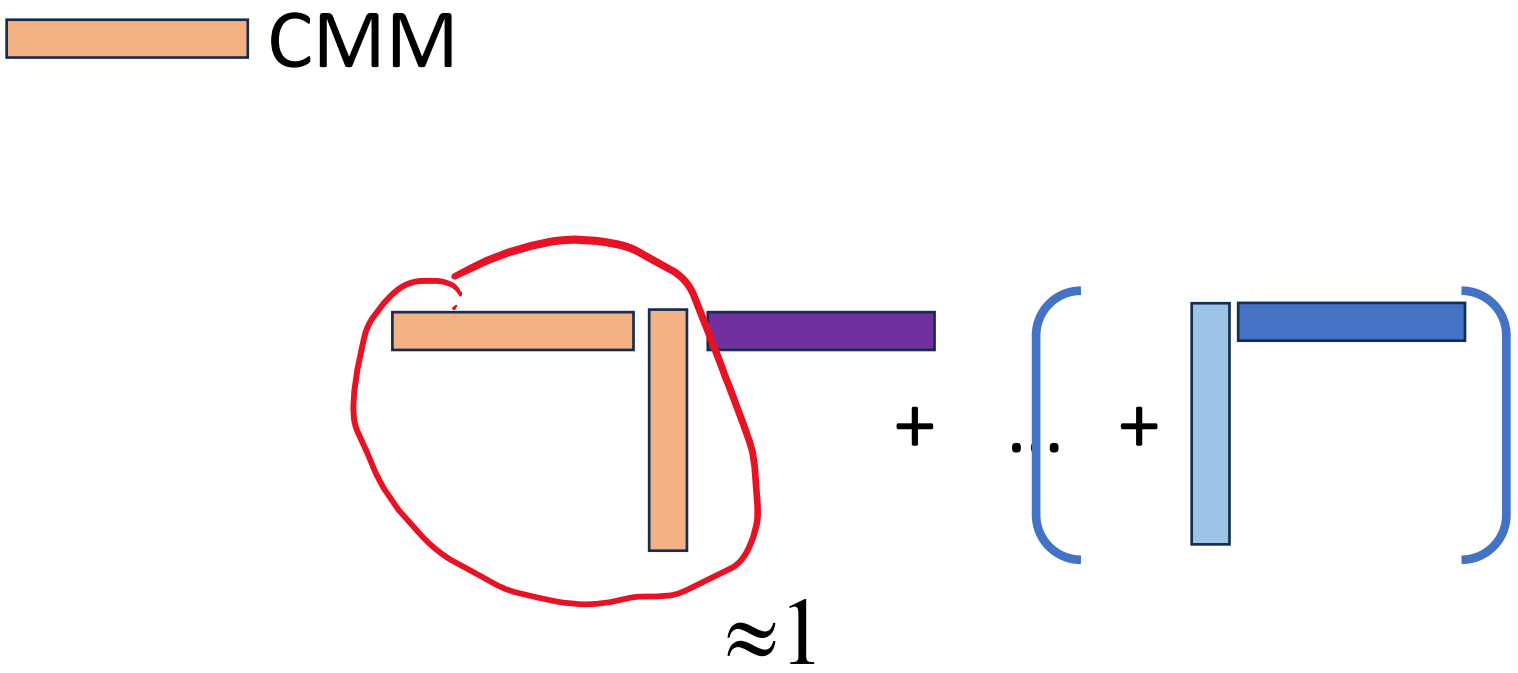
Correlation Matrix Memories (CMM)



Correlation Matrix Memories (CMM)



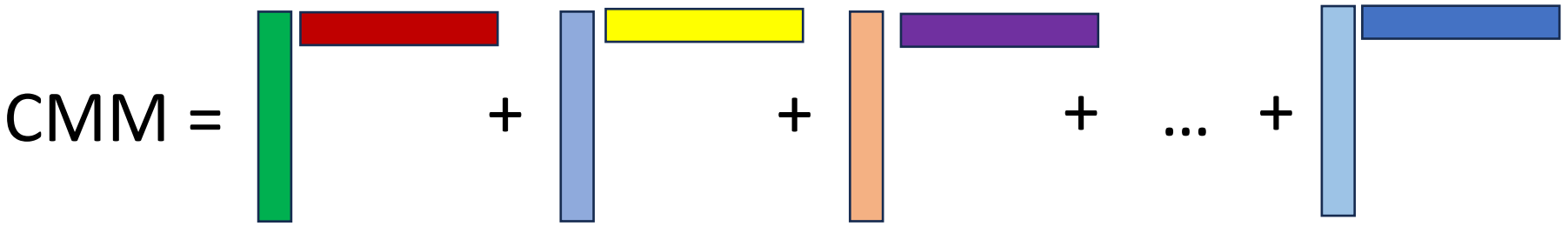
Correlation Matrix Memories (CMM)



Correlation Matrix Memories (CMM)



Correlation Matrix Memories (CMM)





A simple language model (sample space)

He enrolled in the mathematics and physics teaching diploma program.

([He, enrolled, in, the], **mathematics**)

([enrolled, in, the, mathematics], **and**)

([in, the, mathematics, and], **physics**)

([the, mathematics, and, physics], **teaching**)

([mathematics, and, physics, teaching], **diploma**)

([and, physics, teaching, diploma], **program**)

MeMo
Single-layer

Dictionary $w \in D$

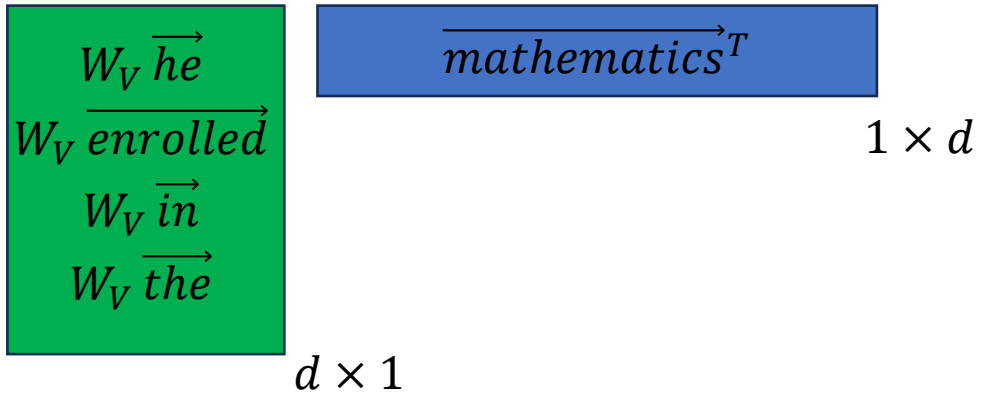
Embedding $\vec{w} \in R^d$

Value Matrix $W_V \frac{d}{h} \times d$

He enrolled in the mathematics and physics teaching diploma program.

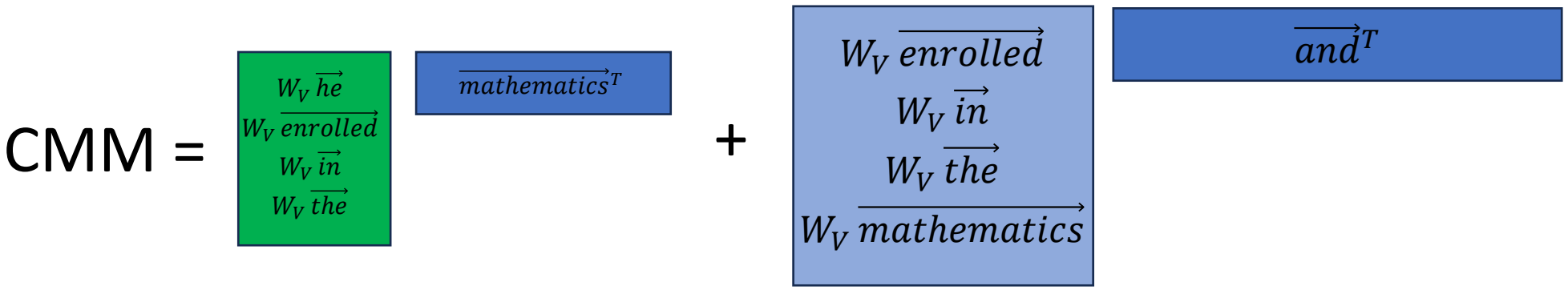
([He, enrolled, in, the], **mathematics**)

CMM =



([He, enrolled, in, the], **mathematics**)

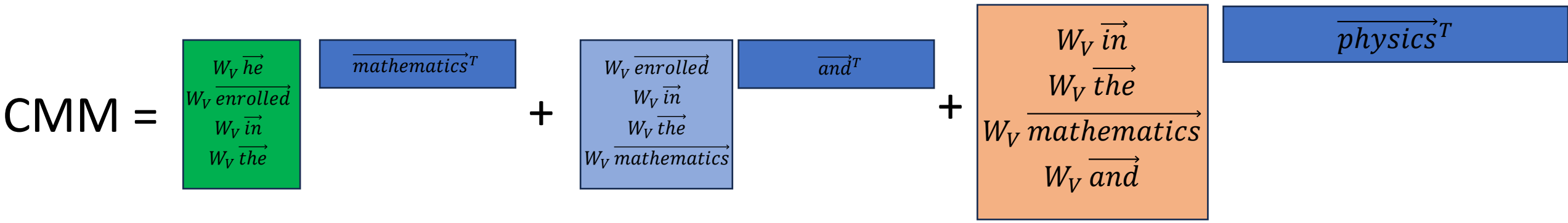
([enrolled, in, the, mathematics], **and**)



([He, enrolled, in, the], **mathematics**)

([enrolled, in, the, mathematics], **and**)

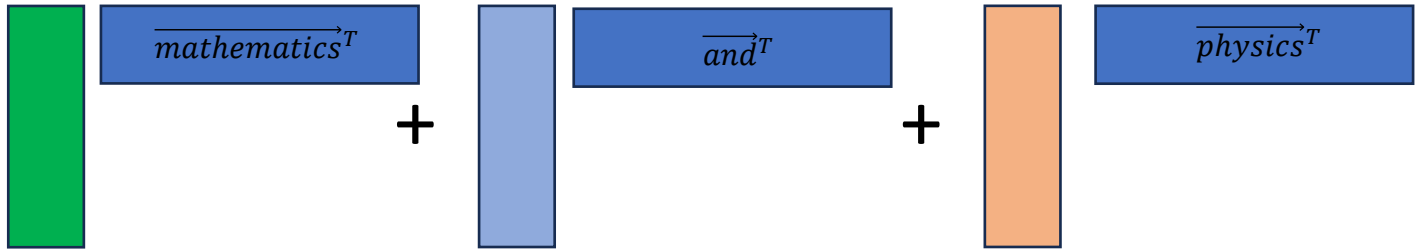
([in, the, mathematics, and], **physics**)



([He, enrolled, in, the], **mathematics**)

([enrolled, in, the, mathematics], **and**)

([in, the, mathematics, and], **physics**)

CMM = 

([He, enrolled, in, the], **mathematics**)

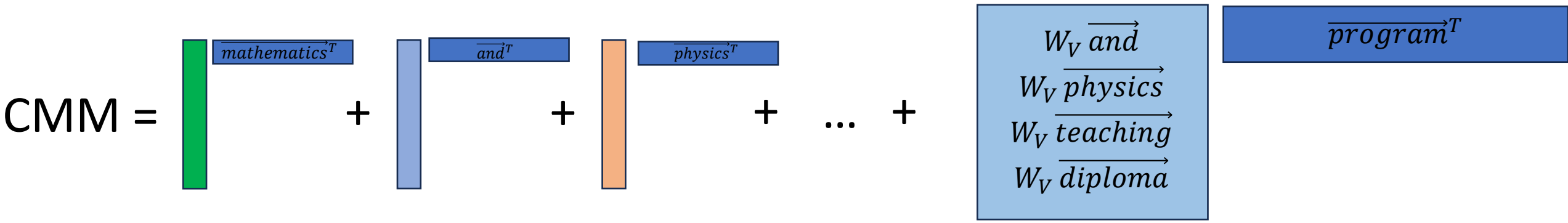
([enrolled, in, the, mathematics], **and**)

([in, the, mathematics, and], **physics**)

([the, mathematics, and, physics], **teaching**)

([mathematics, and, physics, teaching], **diploma**)

([and, physics, teaching, diploma], **program**)



([He, enrolled, in, the], **mathematics**)

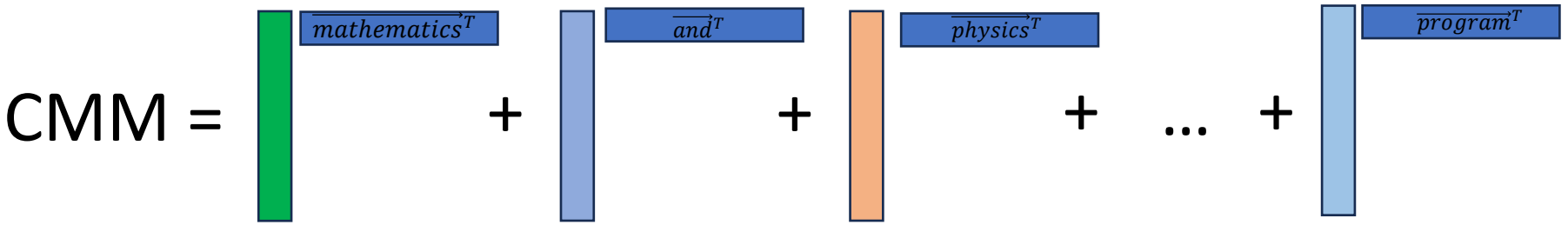
([enrolled, in, the, mathematics], **and**)

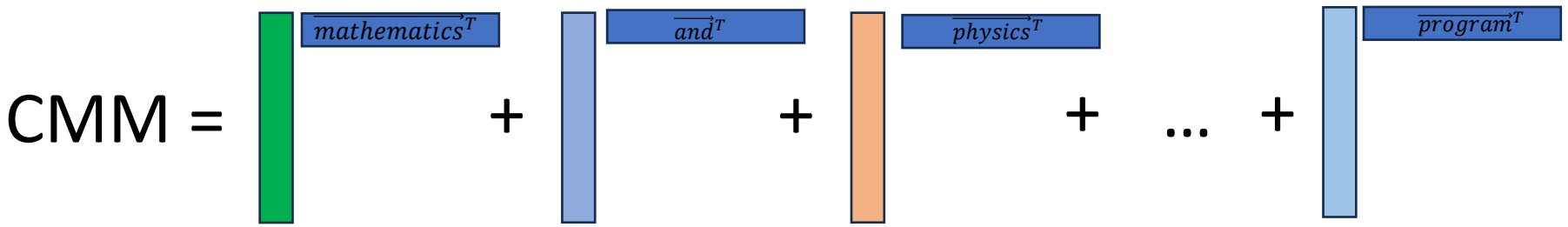
([in, the, mathematics, and], **physics**)

([the, mathematics, and, physics], **teaching**)

([mathematics, and, physics, teaching], **diploma**)

([and, physics, teaching, diploma], **program**)

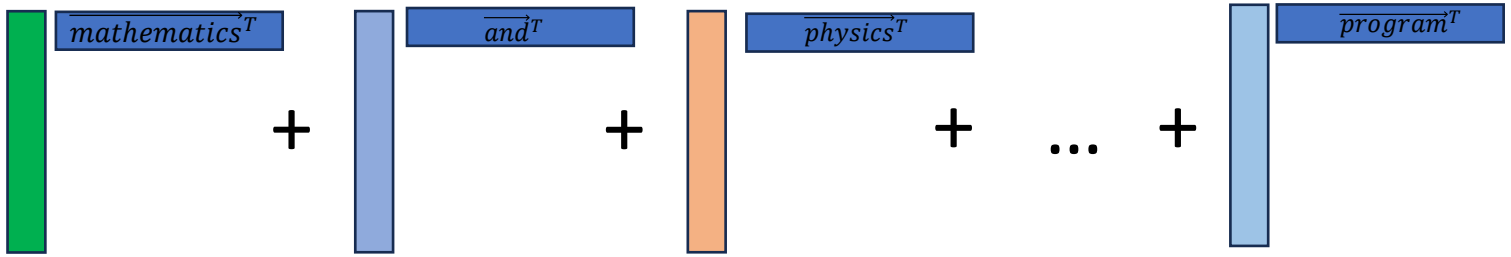




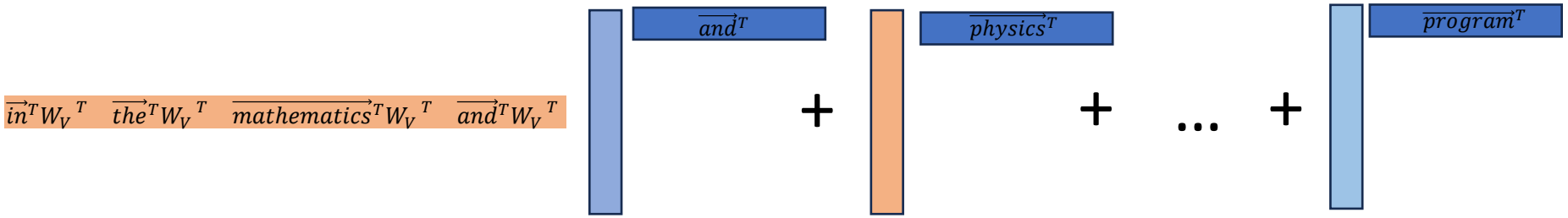
$\overrightarrow{in}^T W_V{}^T$ $\overrightarrow{the}^T W_V{}^T$ $\overrightarrow{mathematics}^T W_V{}^T$ $\overrightarrow{and}^T W_V{}^T$ CMM

CMM =

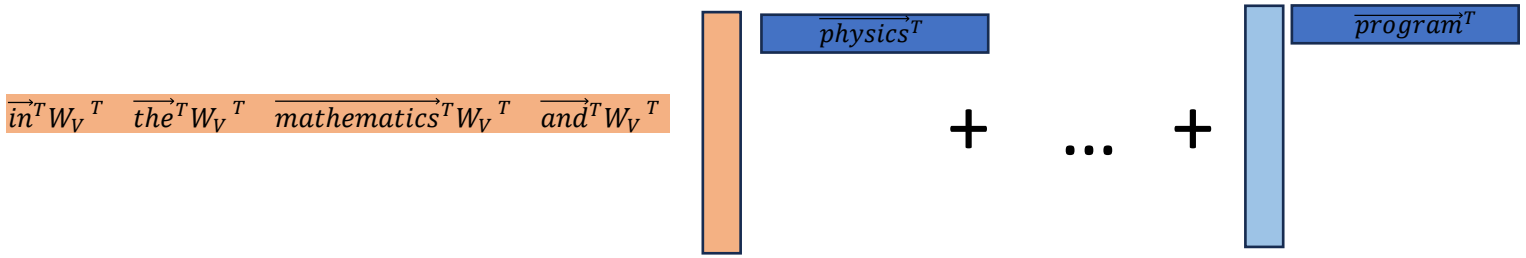
$\overrightarrow{in}^T W_V{}^T \quad \overrightarrow{the}^T W_V{}^T \quad \overrightarrow{mathematics}^T W_V{}^T \quad \overrightarrow{and}^T W_V{}^T$



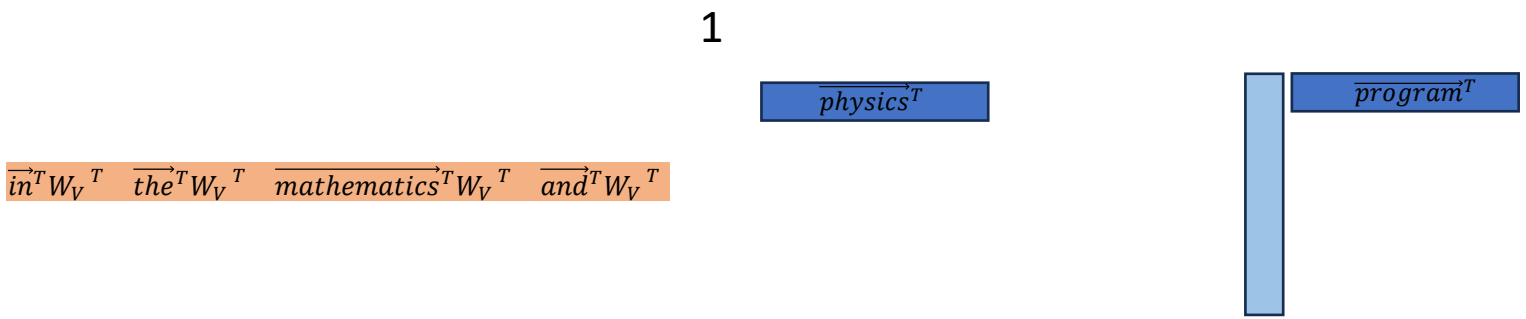
CMM =


$\overrightarrow{in}^T W_V{}^T$ $\overrightarrow{the}^T W_V{}^T$ $\overrightarrow{mathematics}^T W_V{}^T$ $\overrightarrow{and}^T W_V{}^T$
$\overrightarrow{and}^T$
+
$\overrightarrow{physics}^T$
+
...
+
$\overrightarrow{program}^T$

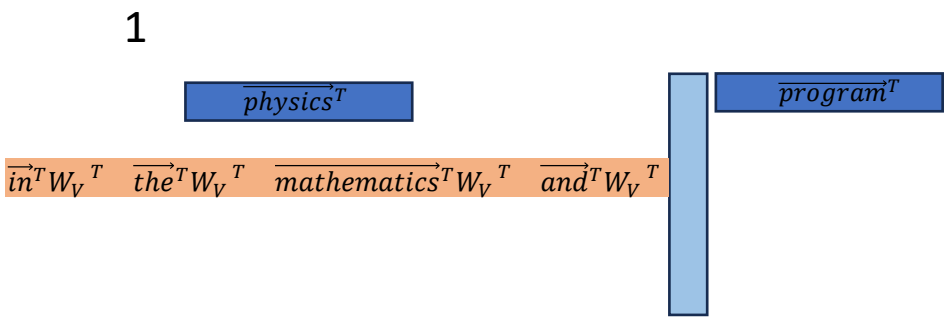
CMM =


$\overrightarrow{in}^T W_V{}^T$ $\overrightarrow{the}^T W_V{}^T$ $\overrightarrow{mathematics}^T W_V{}^T$ $\overrightarrow{and}^T W_V{}^T$
$\overrightarrow{physics}^T$
+ ... +
$\overrightarrow{program}^T$

CMM =


1
$\overrightarrow{physics}^T$
$\overrightarrow{program}^T$
$\overrightarrow{in}^T W_V{}^T$
$\overrightarrow{the}^T W_V{}^T$
$\overrightarrow{mathematics}^T W_V{}^T$
$\overrightarrow{and}^T W_V{}^T$

CMM =

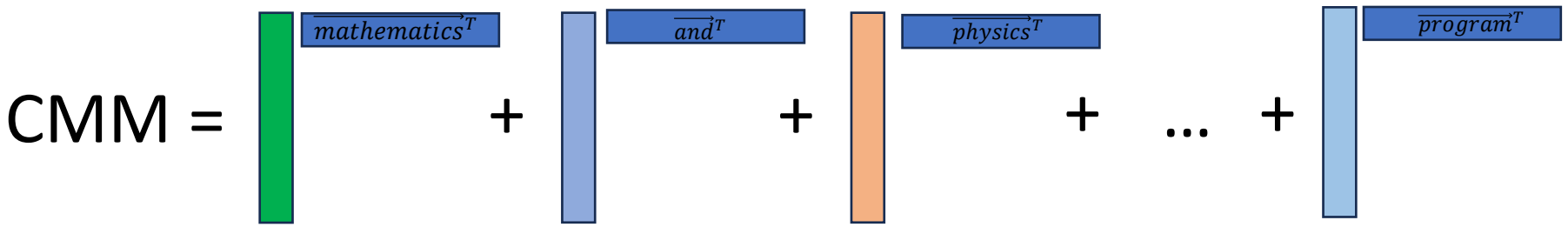

1
$\overrightarrow{physics}^T$
$\overrightarrow{program}^T$
$\overrightarrow{in}^T W_V{}^T$
$\overrightarrow{the}^T W_V{}^T$
$\overrightarrow{mathematics}^T W_V{}^T$
$\overrightarrow{and}^T W_V{}^T$

CMM =

1


$\overrightarrow{physics}^T$


$\overrightarrow{in}^T W_V{}^T$
$\overrightarrow{the}^T W_V{}^T$
$\overrightarrow{mathematics}^T W_V{}^T$
$\overrightarrow{and}^T W_V{}^T$



$\overrightarrow{in}^T W_V{}^T$ $\overrightarrow{the}^T W_V{}^T$ $\overrightarrow{mathematics}^T W_V{}^T$ $\overrightarrow{and}^T W_V{}^T$ CMM = $\overrightarrow{physics}^T$

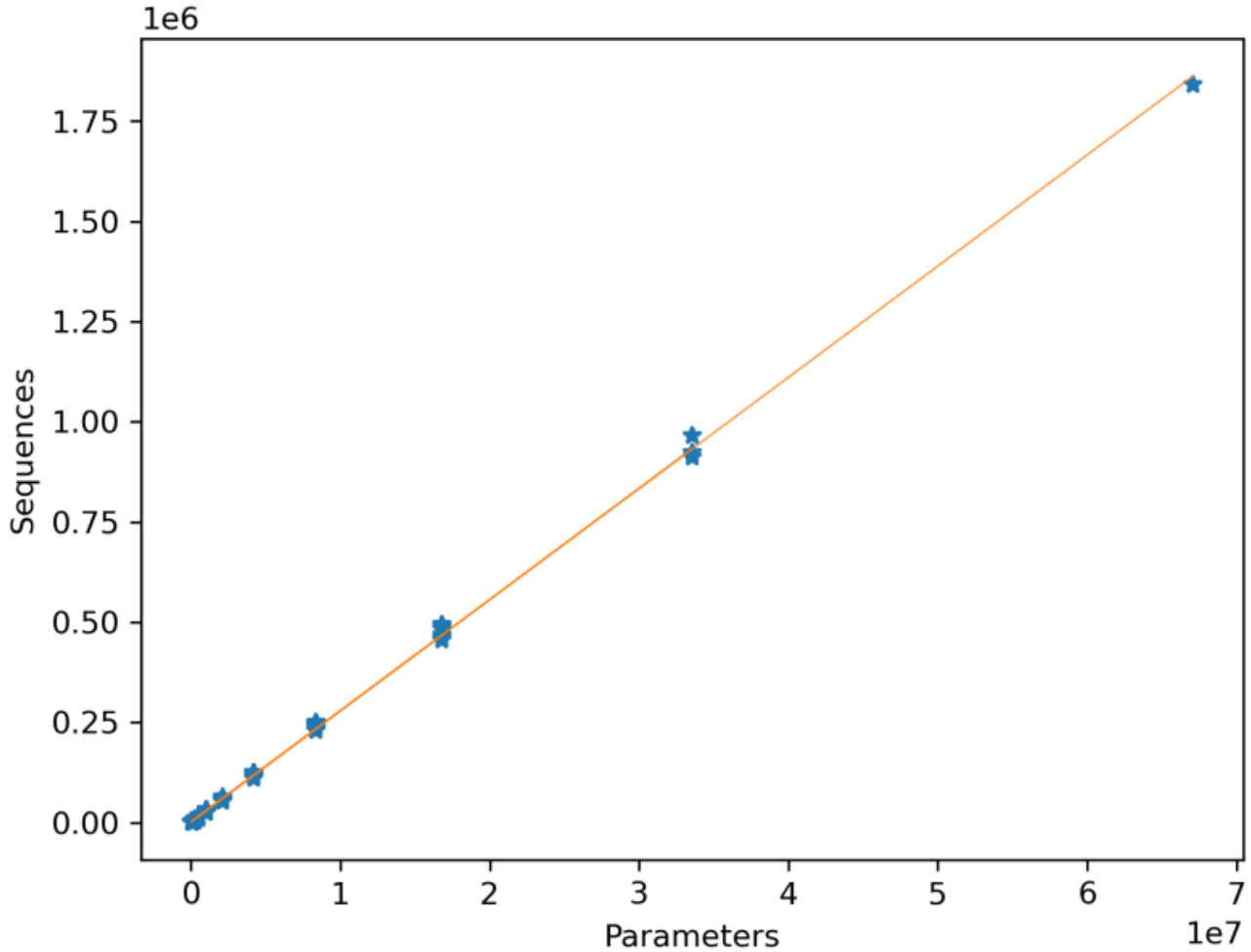
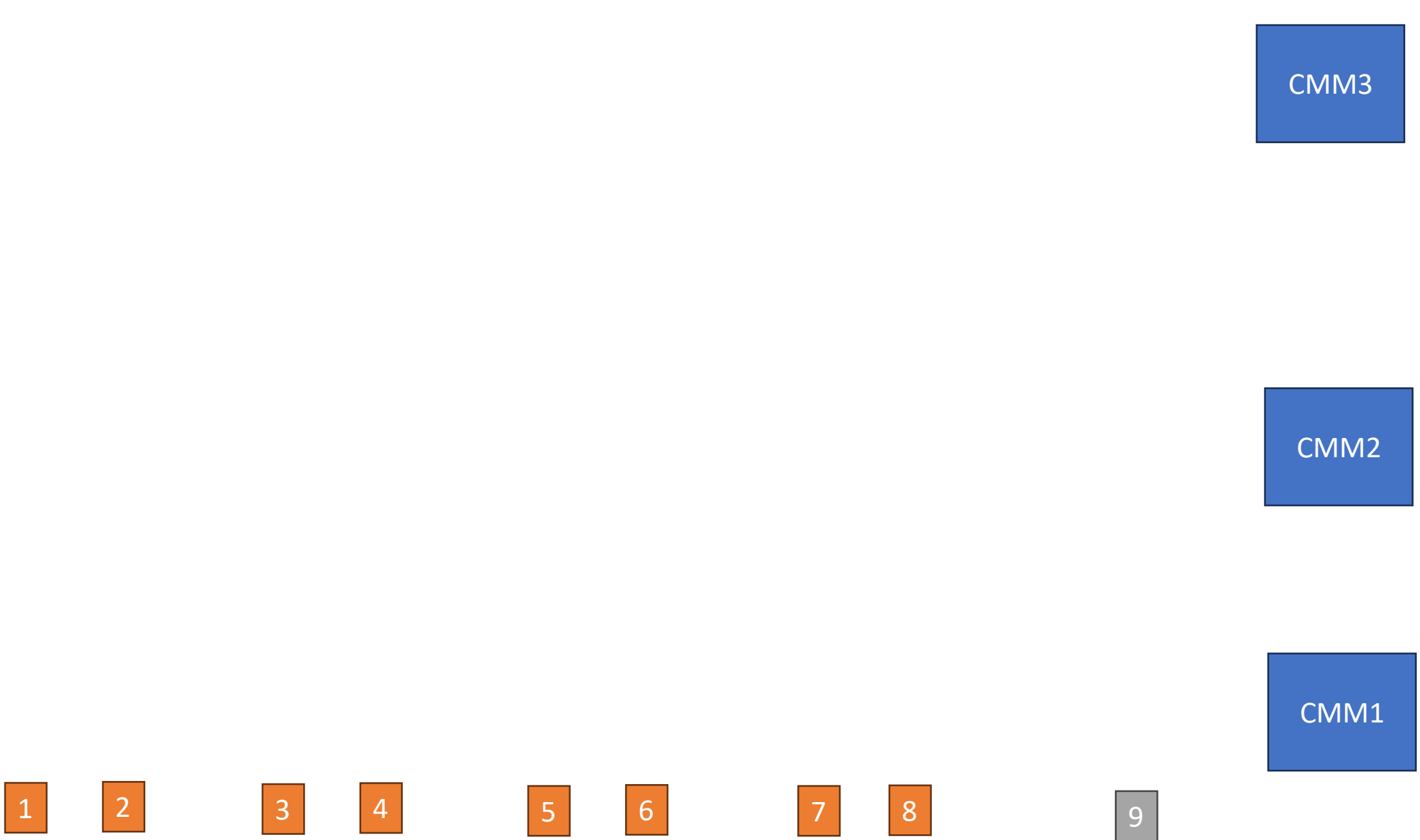
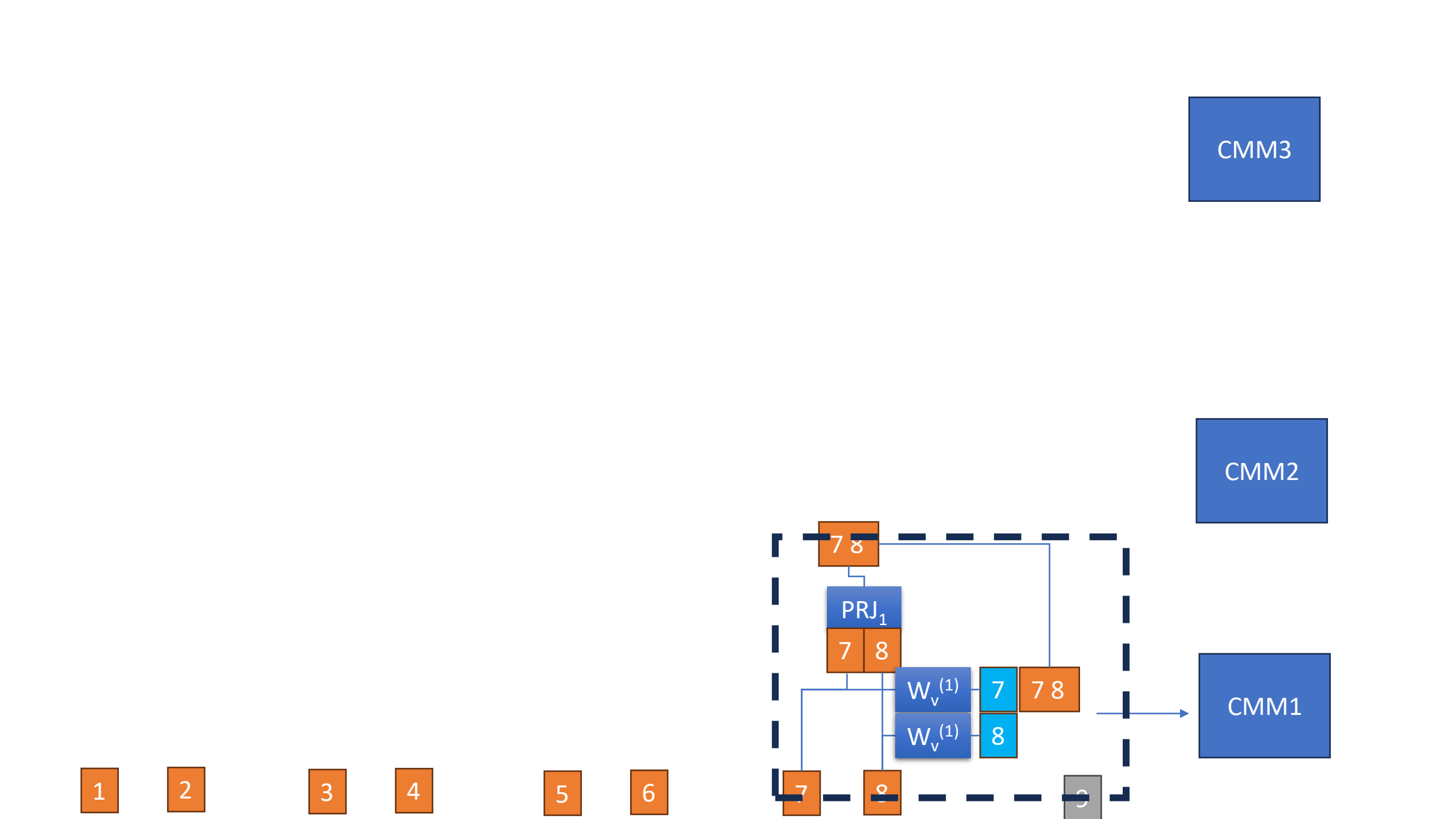


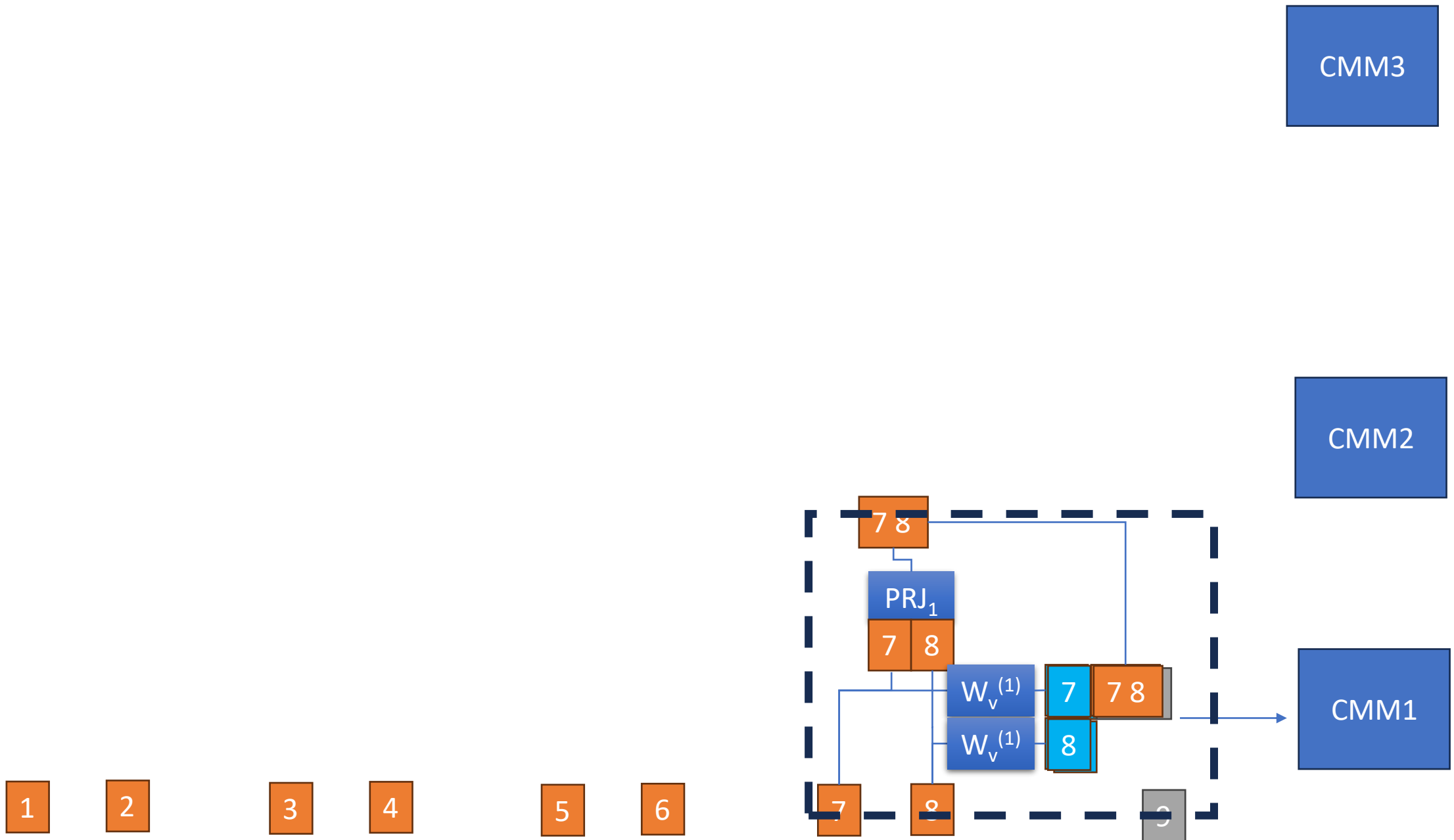
Figure 3: Memorization capacity of a single CMM: parameters $NoP = h \cdot d_h \cdot d$ with respect to the number of sequences that can be stored. Points in the plot are CMMs with different configurations of h, d_h, and d.

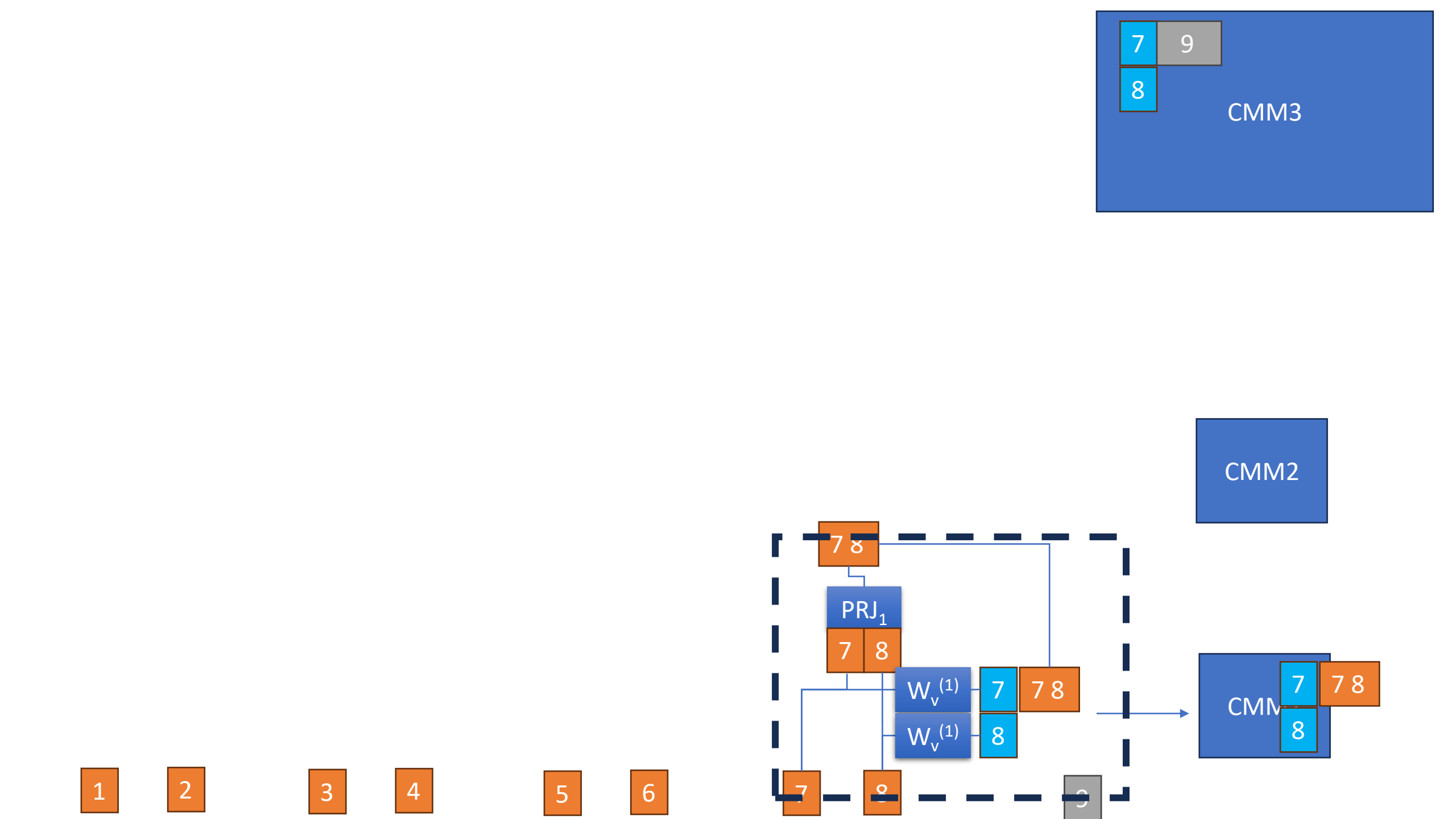
MeMo
Multi-layer

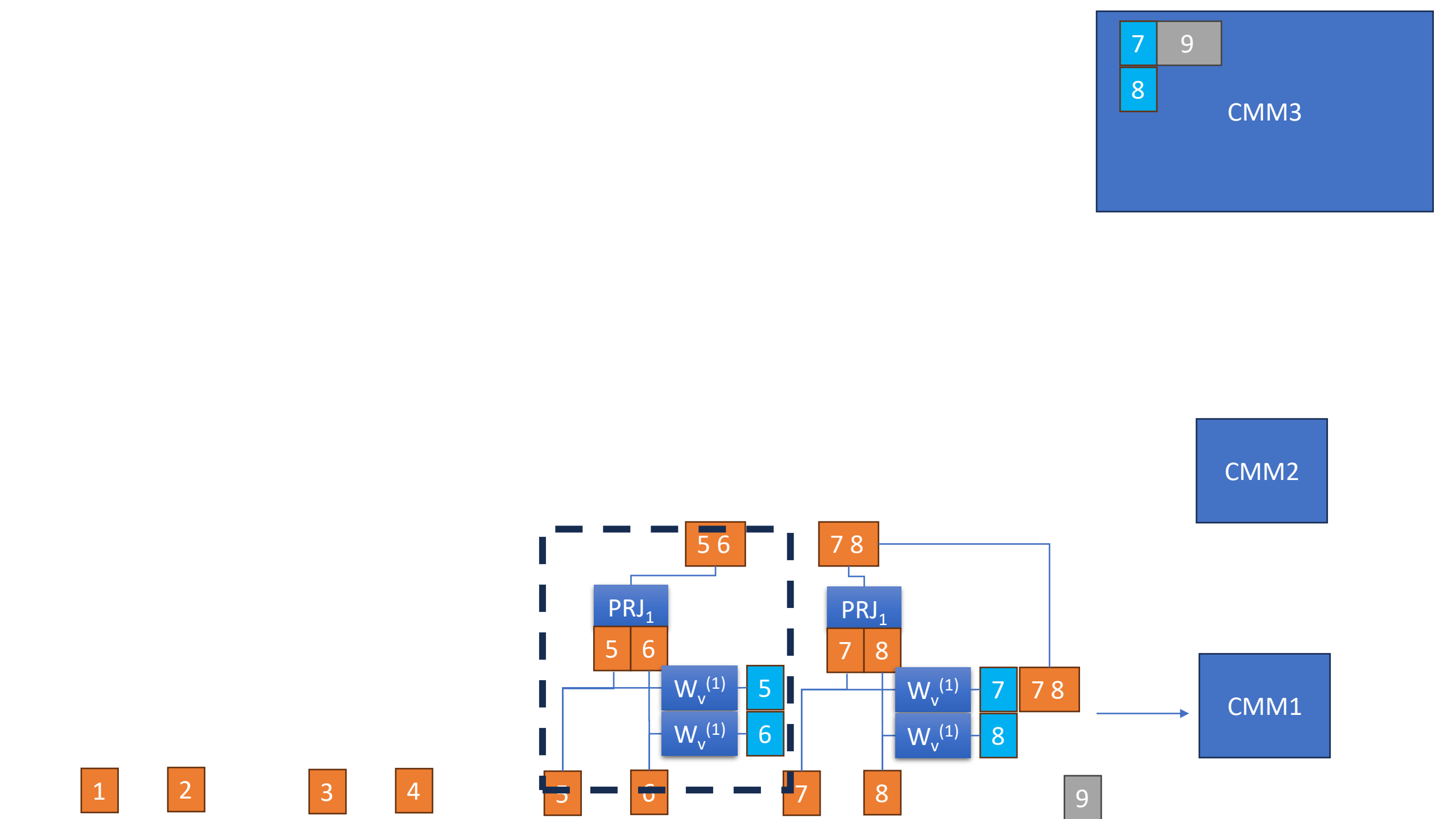
Memorization

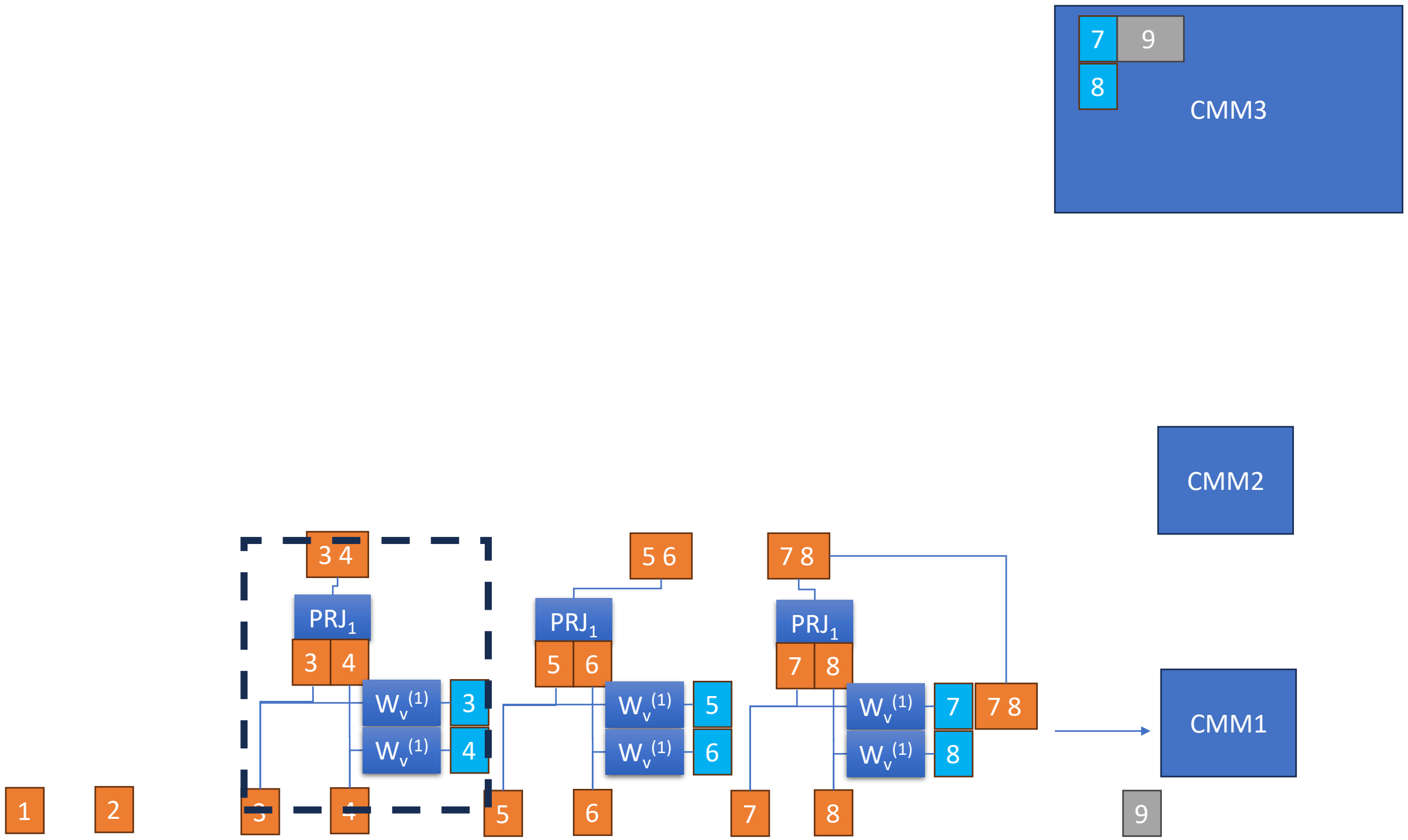

CMM3
CMM2
CMM1
1
2
3
4
5
6
7
8
9

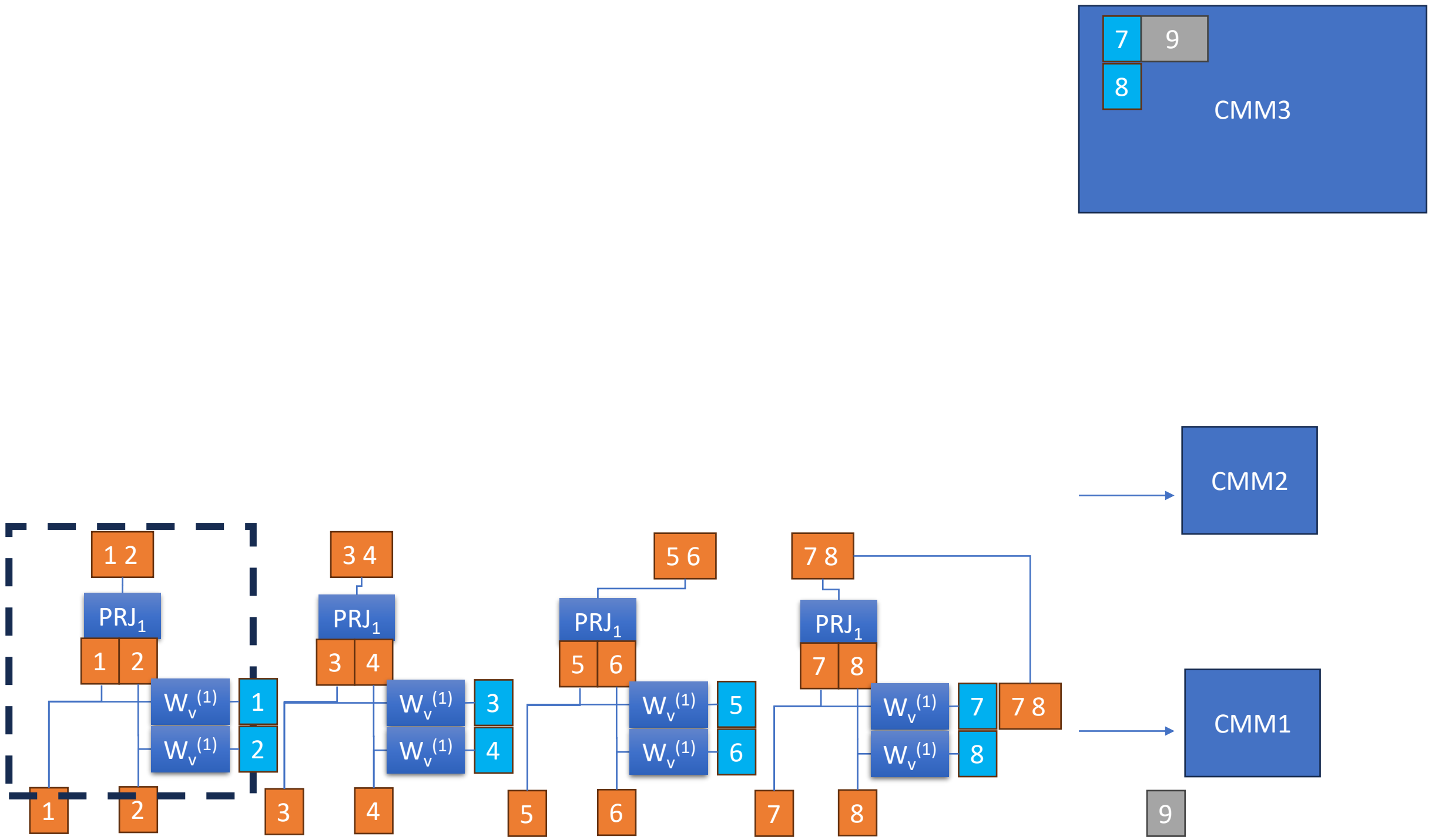

CMM3
CMM2
CMM1
7 8
PRJ1
7
8
Wv(1)
Wv(1)
7
8
7 8
1
2
3
4
5
6
7
8
9

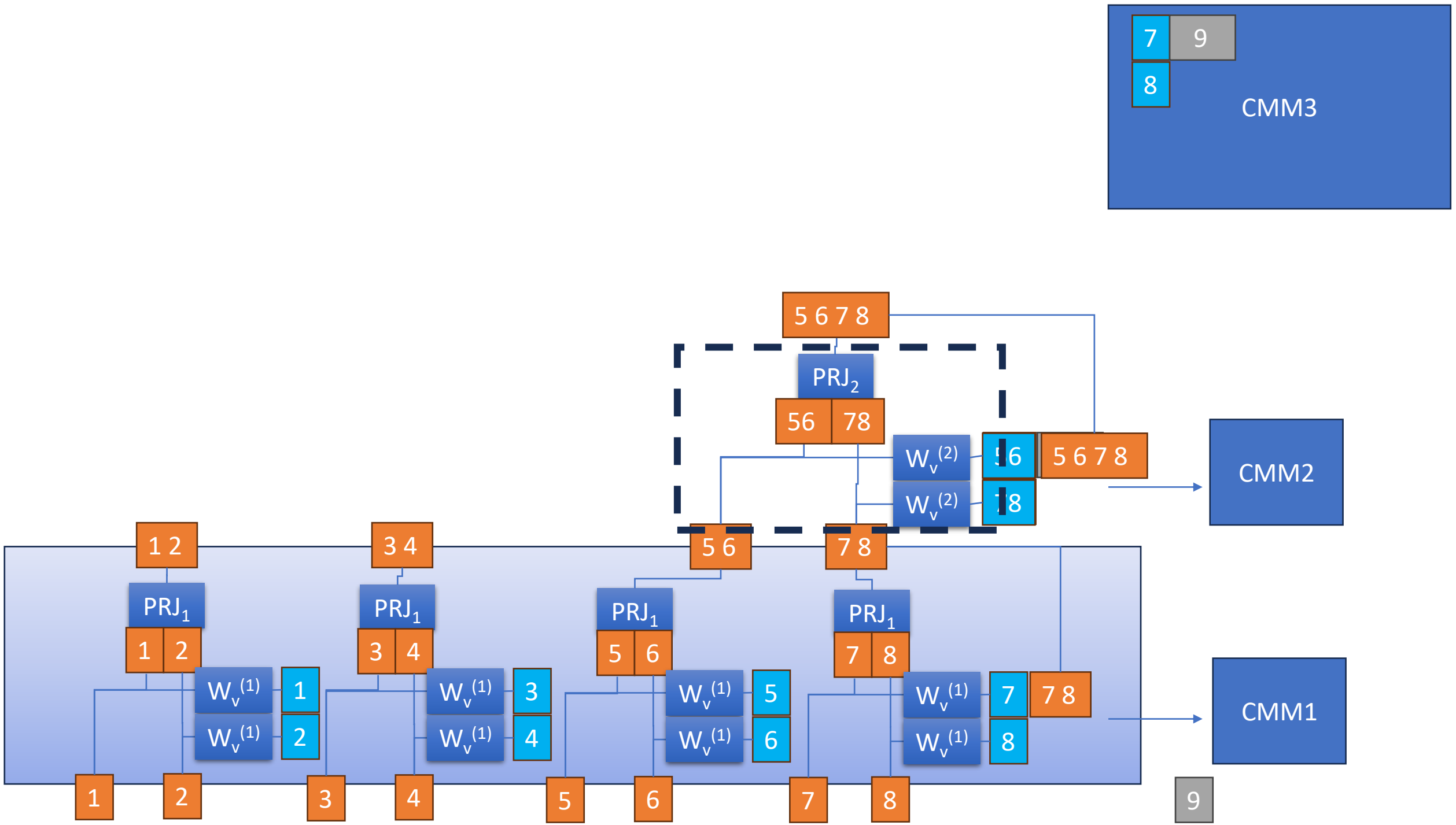

CMM3
CMM2
CMM1
7 8
PRJ1
7
8
Wv(1)
Wv(1)
7
7 8
8
1
2
3
4
5
6
7
8
9

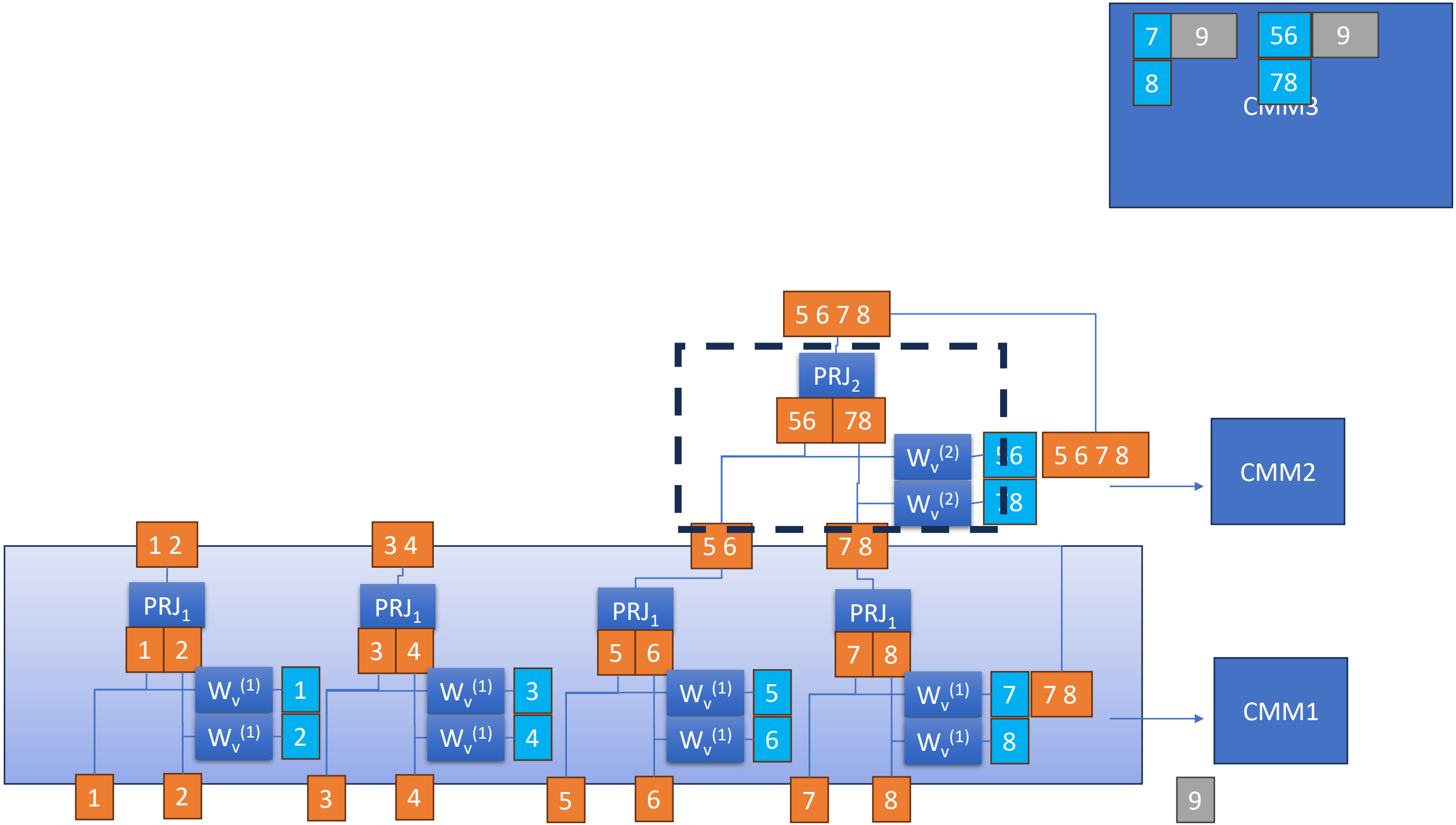

7
9
8
CMM3
CMM2
7 8
PRJ1
7
8
Wv(1)
Wv(1)
7
8
7 8
CMM
7
7 8
8
1
2
3
4
5
6
7
8
9

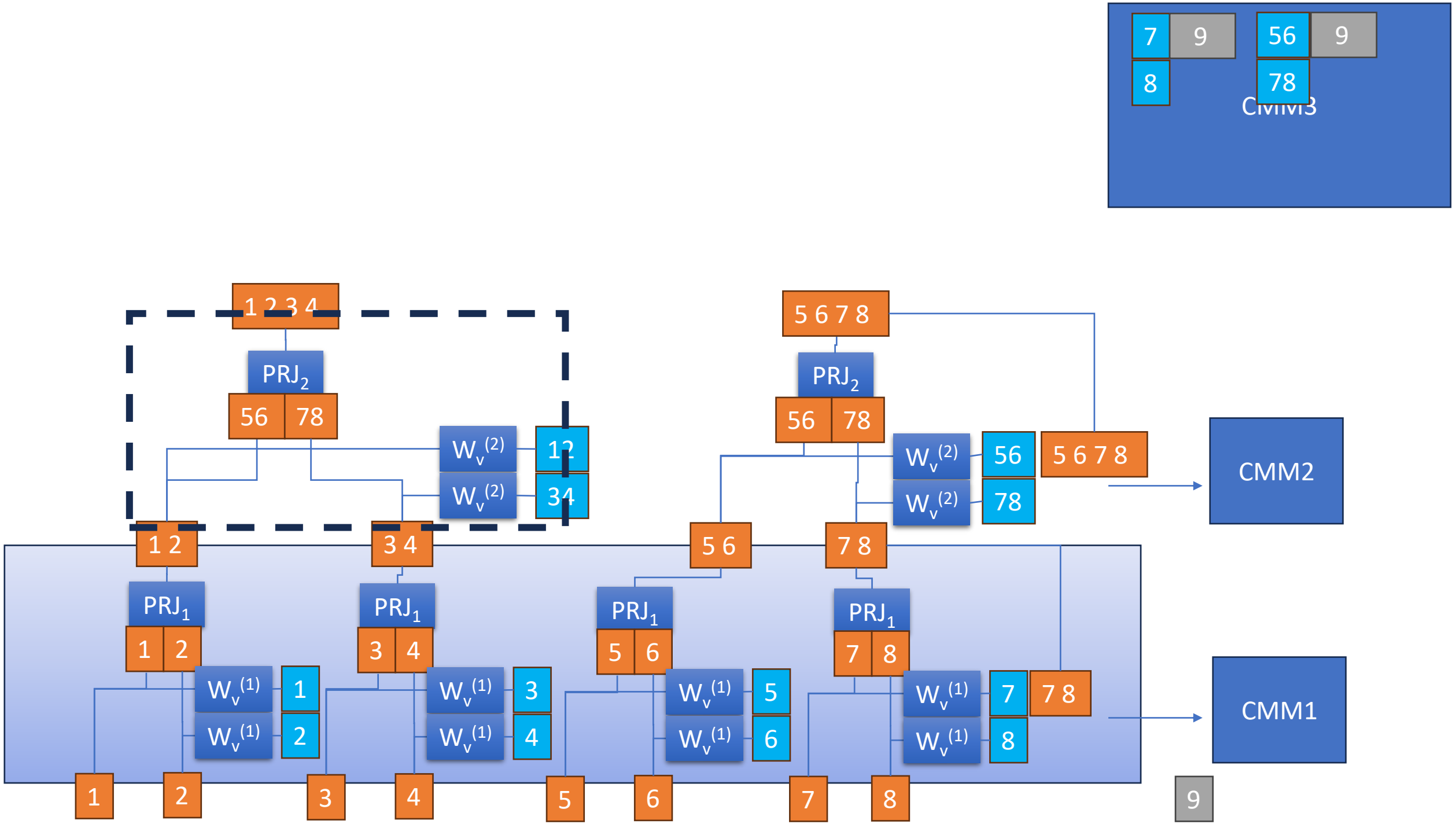

7
9
8
CMM3
CMM2
5 6
7 8
PRJ1
5
6
PRJ1
7
8
Wv(1)
5
Wv(1)
6
Wv(1)
7
7 8
Wv(1)
8
CMM1
1
2
3
4
5
6
7
8
9

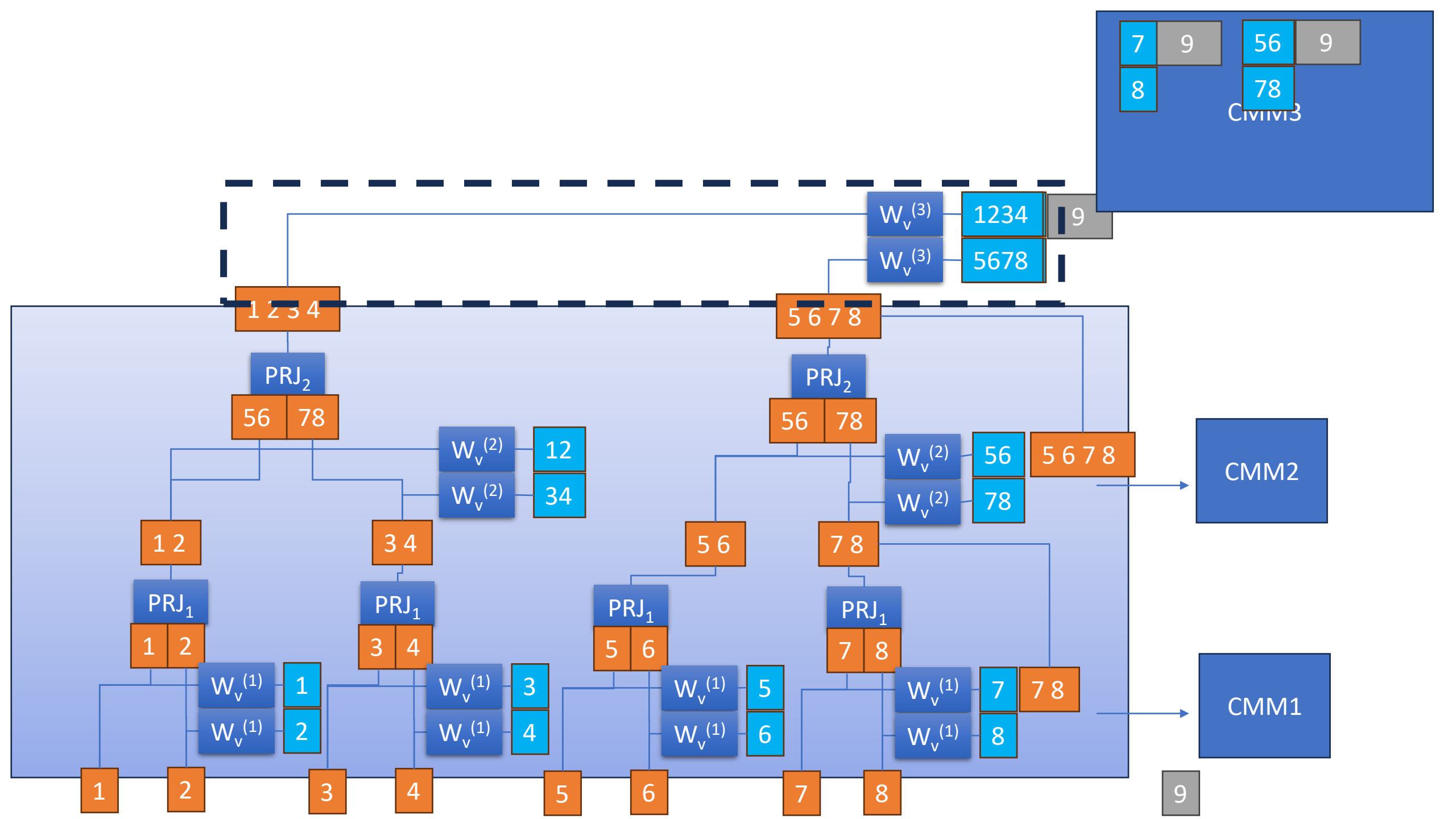

7
9
8
CMM3
CMM2
3 4
PRJ1
3
4
Wv(1)
3
Wv(1)
4
5 6
PRJ1
5
6
Wv(1)
5
Wv(1)
6
7 8
PRJ1
7
8
Wv(1)
7
7 8
Wv(1)
8
CMM1
1
2
3
4
5
6
7
8
9

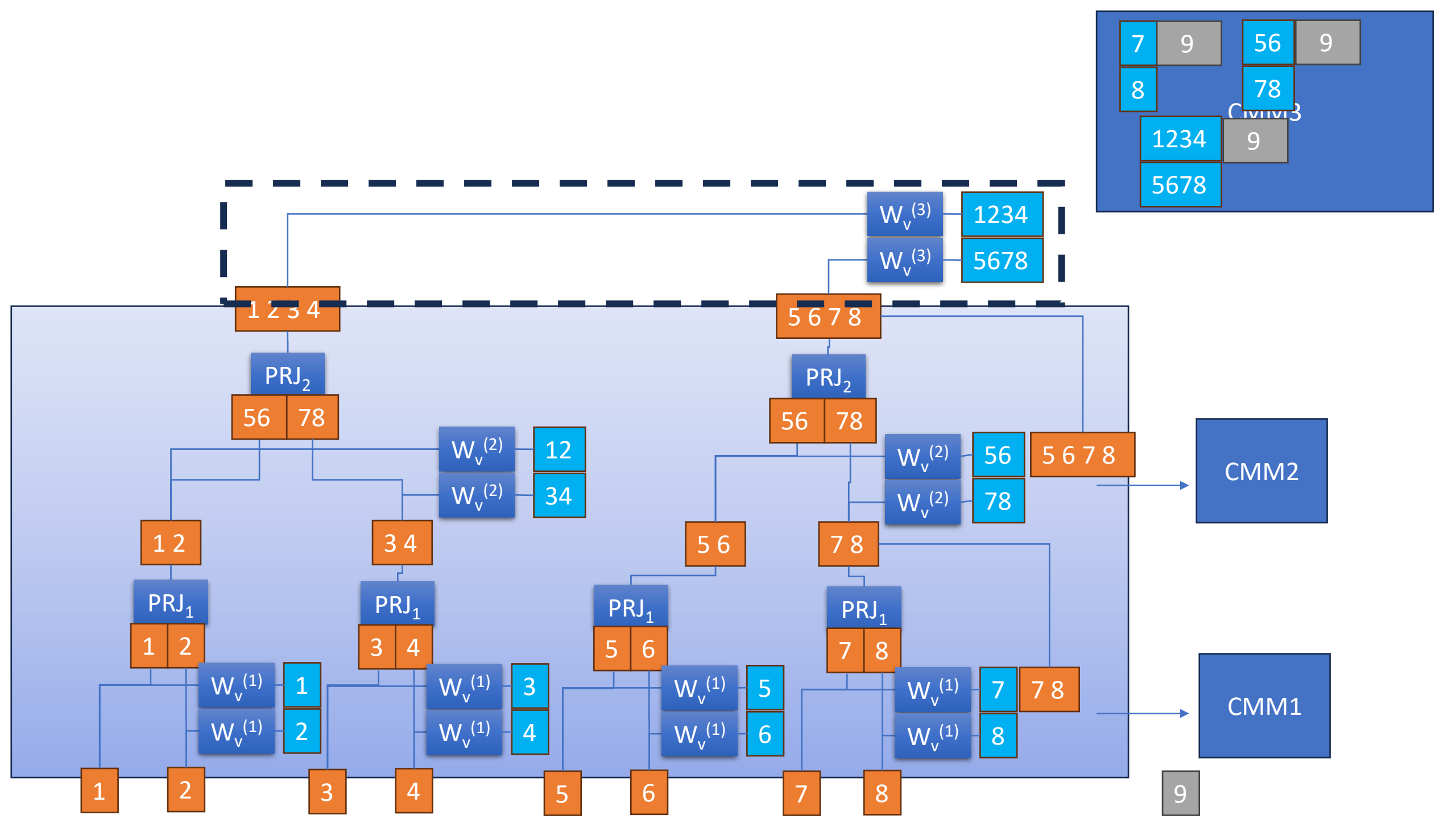

7
9
8
CMM3
CMM2
CMM1
1 2
3 4
5 6
7 8
PRJ1
W_v^(1)
1
2
3
4
5
6
7
8
9

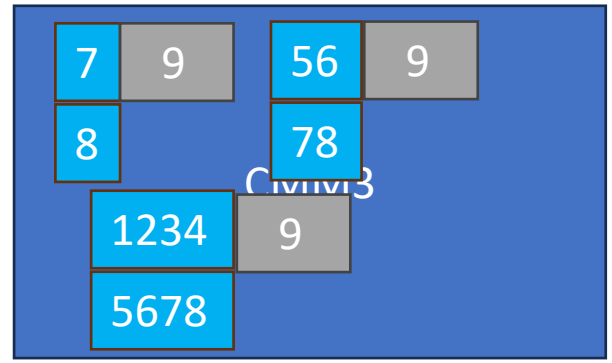

7
9
8
CMM3
5 6 7 8
PRJ2
56
78
W v (2)
W v (2)
5 6 7 8
CMM2
1 2
3 4
5 6
7 8
PRJ1
PRJ1
PRJ1
PRJ1
1
2
3
4
5
6
7
8
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
7 8
CMM1
9

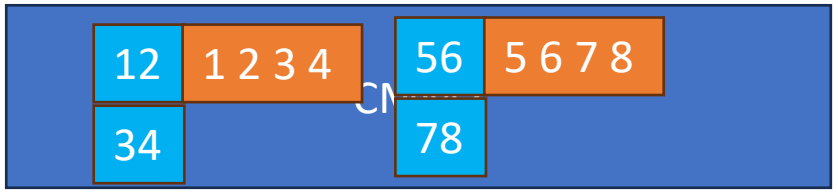

7
9
8
56
9
78
CMM3
5 6 7 8
PRJ2
56
78
W v (2)
W v (2)
5 6 7 8
CMM2
1 2
3 4
5 6
7 8
PRJ1
PRJ1
PRJ1
PRJ1
1 2
3 4
5 6
7 8
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
W v (1)
1
2
3
4
5
6
7
8
7 8
CMM1
1
2
3
4
5
6
7
8
9

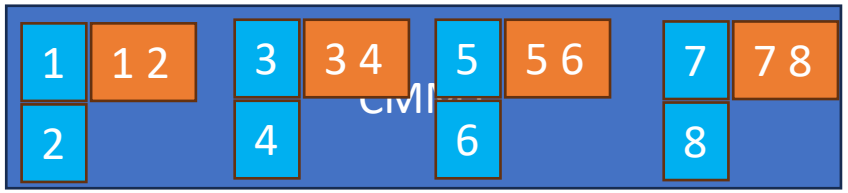

7
9
8
56
9
78
CMM3
1 2 3 4
PRJ2
56
78
W v (2)
W v (2)
12
34
1 2
3 4
PRJ1
PRJ1
1
2
3
4
W v (1)
W v (1)
W v (1)
W v (1)
1
2
3
4
5 6 7 8
PRJ2
56
78
W v (2)
W v (2)
56
78
5 6 7 8
CMM2
5 6
7 8
PRJ1
PRJ1
5
6
7
8
W v (1)
W v (1)
W v (1)
W v (1)
5
6
7
8
7 8
CMM1
9


7
9
8
56
9
78
CMM3
$W_v^{(3)}$
1234
9
$W_v^{(3)}$
5678
1 2 3 4
5 6 7 8
PRJ_2
56
78
PRJ_2
56
78
$W_v^{(2)}$
12
$W_v^{(2)}$
34
$W_v^{(2)}$
56
$W_v^{(2)}$
78
5 6 7 8
CMM2
1 2
3 4
5 6
7 8
PRJ_1
1
2
PRJ_1
3
4
PRJ_1
5
6
PRJ_1
7
8
$W_v^{(1)}$
1
$W_v^{(1)}$
2
$W_v^{(1)}$
3
$W_v^{(1)}$
4
$W_v^{(1)}$
5
$W_v^{(1)}$
6
$W_v^{(1)}$
7
$W_v^{(1)}$
8
7 8
CMM1
1
2
3
4
5
6
7
8
9

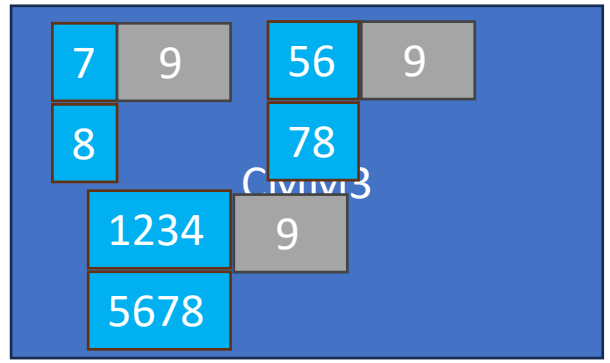

7
9
56
9
8
78
1234
9
5678
CMM3
$W_v^{(3)}$
1234
$W_v^{(3)}$
5678
1 2 3 4
5 6 7 8
PRJ_2
56
78
$W_v^{(2)}$
12
$W_v^{(2)}$
34
PRJ_2
56
78
$W_v^{(2)}$
56
$W_v^{(2)}$
78
5 6 7 8
CMM2
1 2
3 4
5 6
7 8
PRJ_1
1
2
PRJ_1
3
4
PRJ_1
5
6
PRJ_1
7
8
$W_v^{(1)}$
1
$W_v^{(1)}$
2
$W_v^{(1)}$
3
$W_v^{(1)}$
4
$W_v^{(1)}$
5
$W_v^{(1)}$
6
$W_v^{(1)}$
7
$W_v^{(1)}$
8
7 8
CMM1
1
2
3
4
5
6
7
8
9

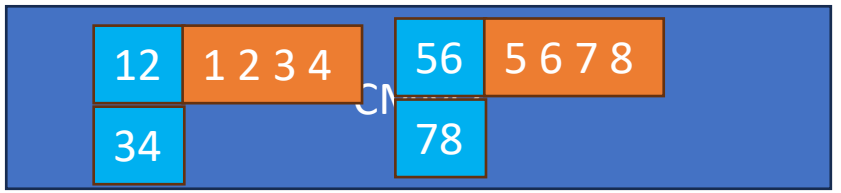

7
9
56
9
8
78
1234
9
5678

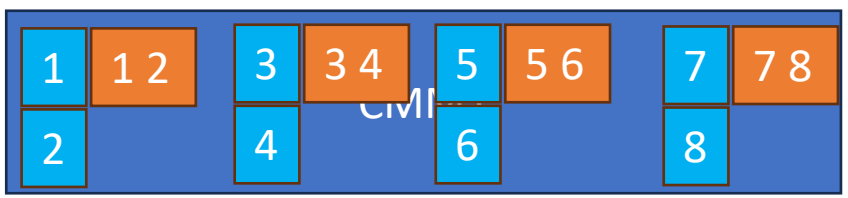

12
1 2 3 4
56
5 6 7 8
34
78


1
1 2
3
3 4
5
5 6
7
7 8
2
4
6
8

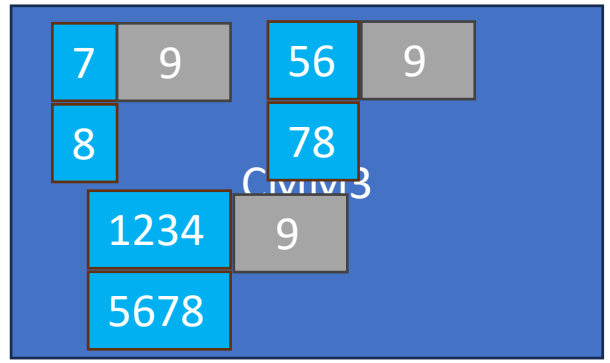
Retreiving

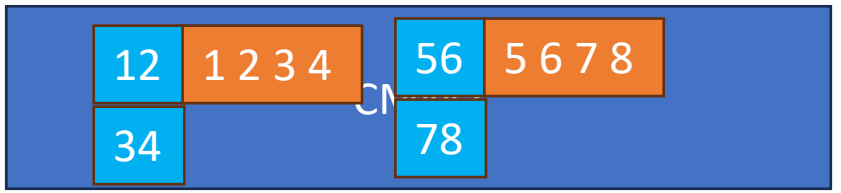

7
9
56
9
8
78
1234
9
5678

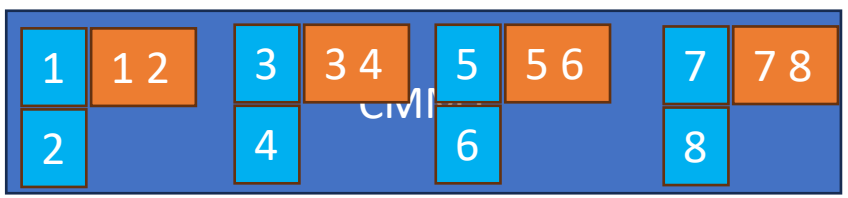

12
1 2 3 4
56
5 6 7 8
34
78


1
1 2
3
3 4
5
5 6
7
7 8
2
4
6
8

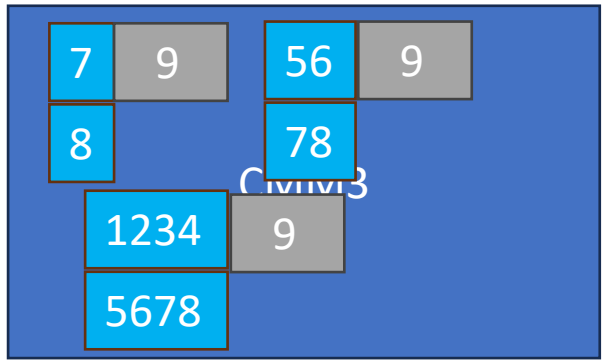

1
2
3
4
5
6
7
8

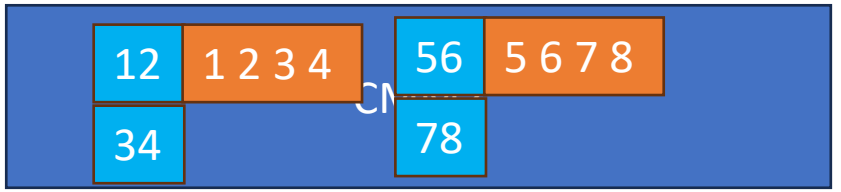

7
9
56
9
8
78
1234
9
5678

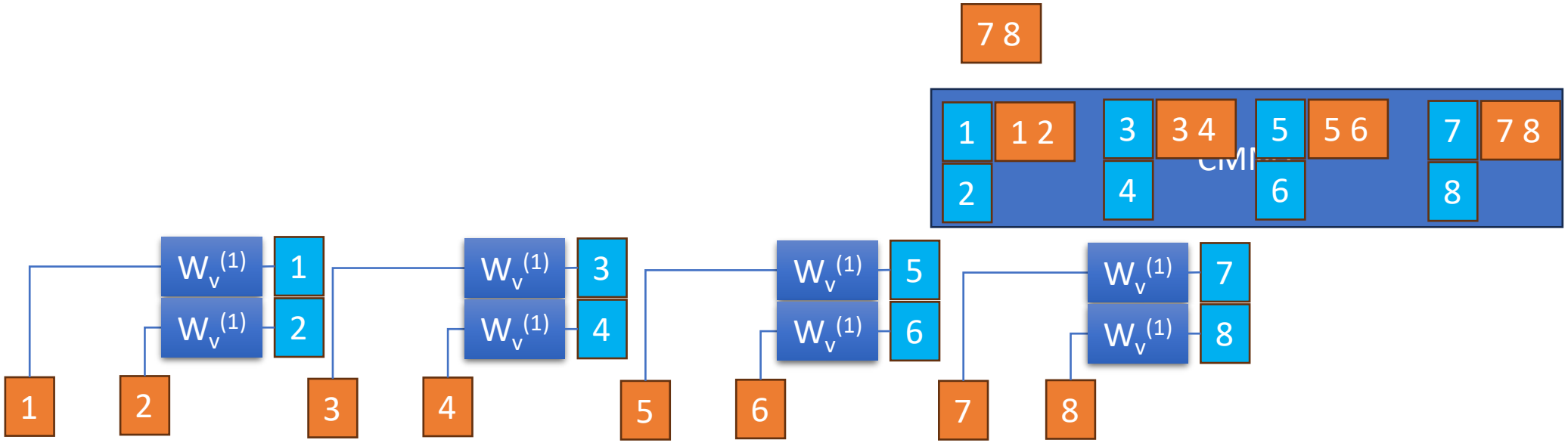

12
1 2 3 4
56
5 6 7 8
34
78

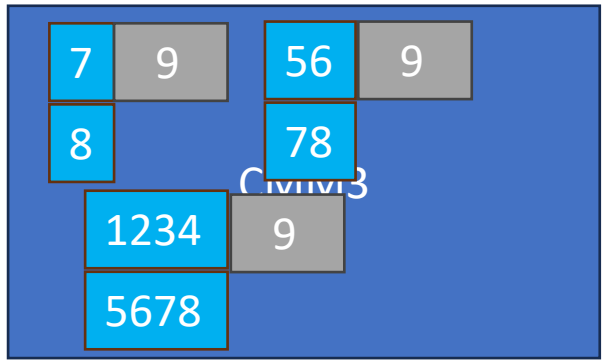

1
1 2
3
3 4
5
5 6
7
7 8
2
4
6
8

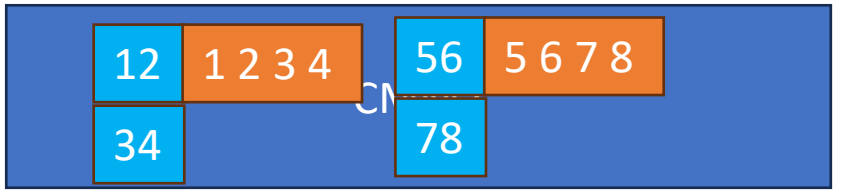

1
2
3
4
5
6
7
8

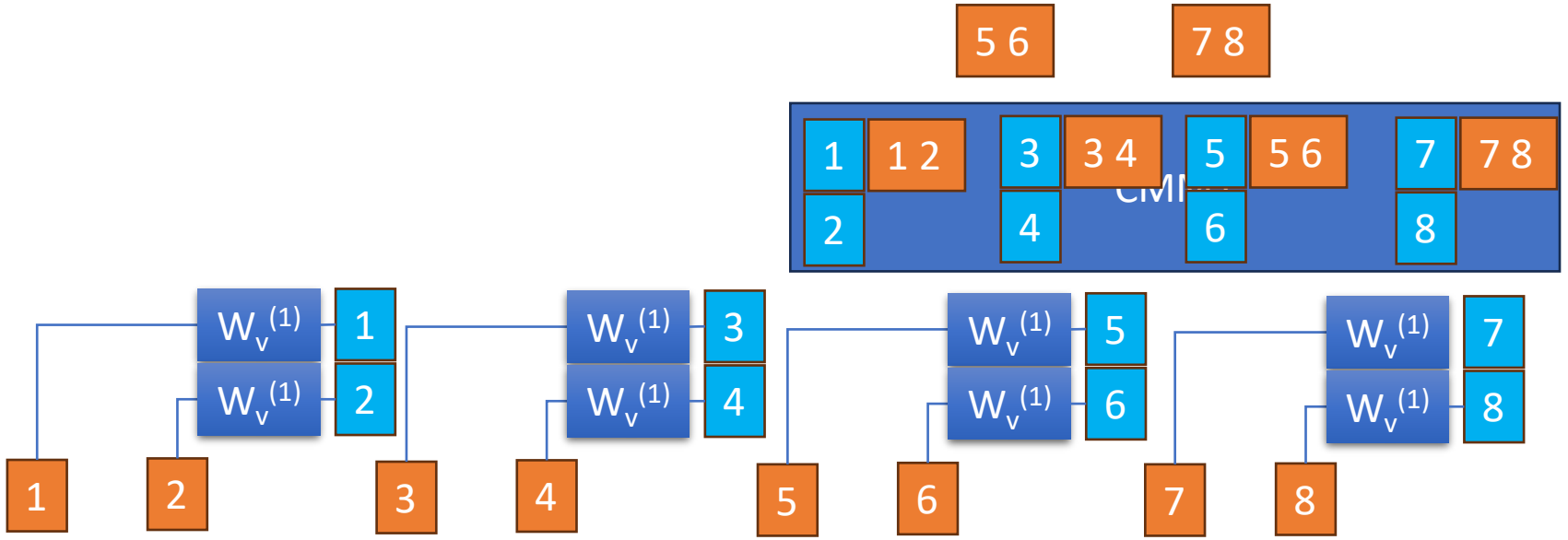

7
9
56
9
8
78
1234
9
5678

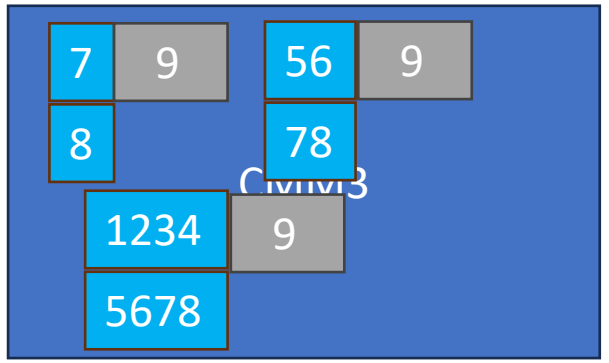

12
1 2 3 4
56
5 6 7 8
34
78

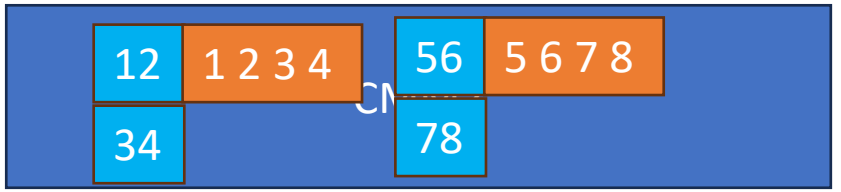

7 8
1
1 2
3
3 4
5
5 6
7
7 8
2
4
6
8
$W_v^{(1)}$
1
$W_v^{(1)}$
2
$W_v^{(1)}$
3
$W_v^{(1)}$
4
$W_v^{(1)}$
5
$W_v^{(1)}$
6
$W_v^{(1)}$
7
$W_v^{(1)}$
8
1
2
3
4
5
6
7
8

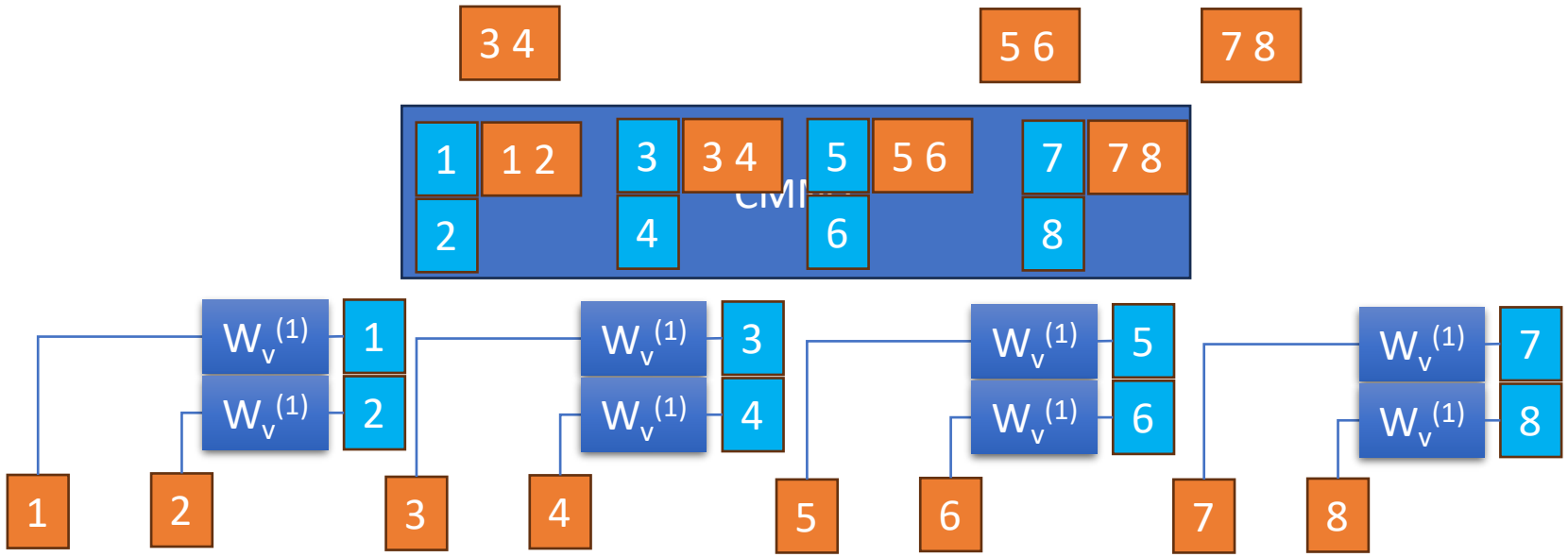

7
9
56
9
8
78
1234
9
5678

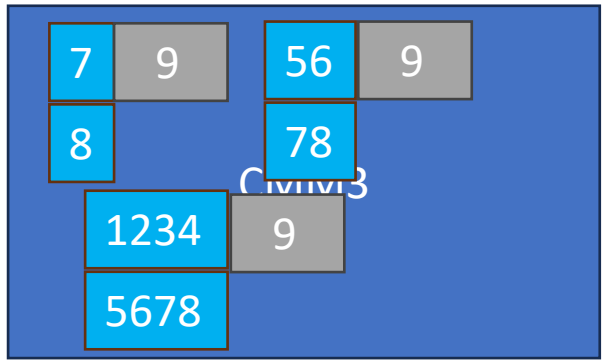

12
1 2 3 4
56
5 6 7 8
34
78

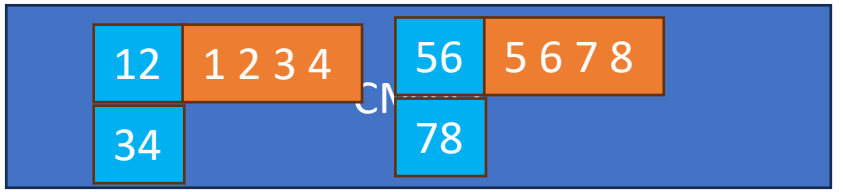

5 6
7 8
1
1 2
3
3 4
5
5 6
7
7 8
2
4
6
8
$W_v^{(1)}$
1
$W_v^{(1)}$
2
$W_v^{(1)}$
3
$W_v^{(1)}$
4
$W_v^{(1)}$
5
$W_v^{(1)}$
6
$W_v^{(1)}$
7
$W_v^{(1)}$
8
1
2
3
4
5
6
7
8

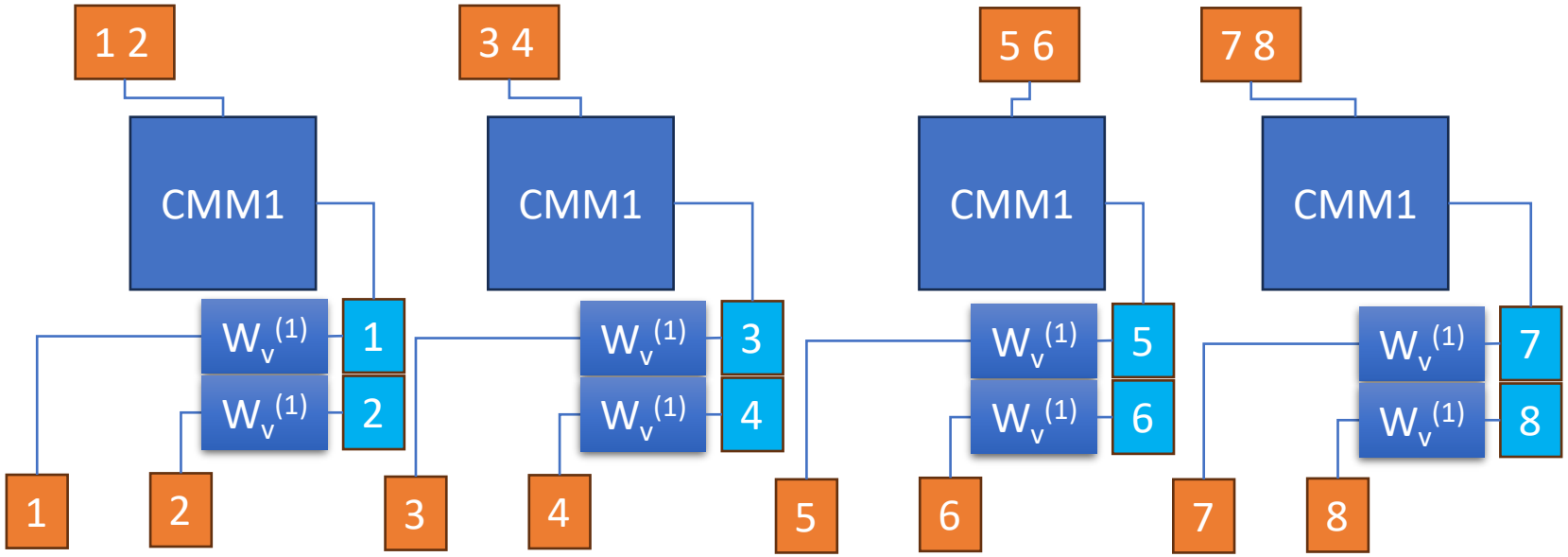

7
9
56
9
8
78
1234
9
5678

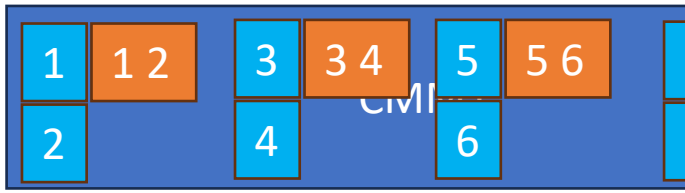

12
1 2 3 4
56
5 6 7 8
34
78

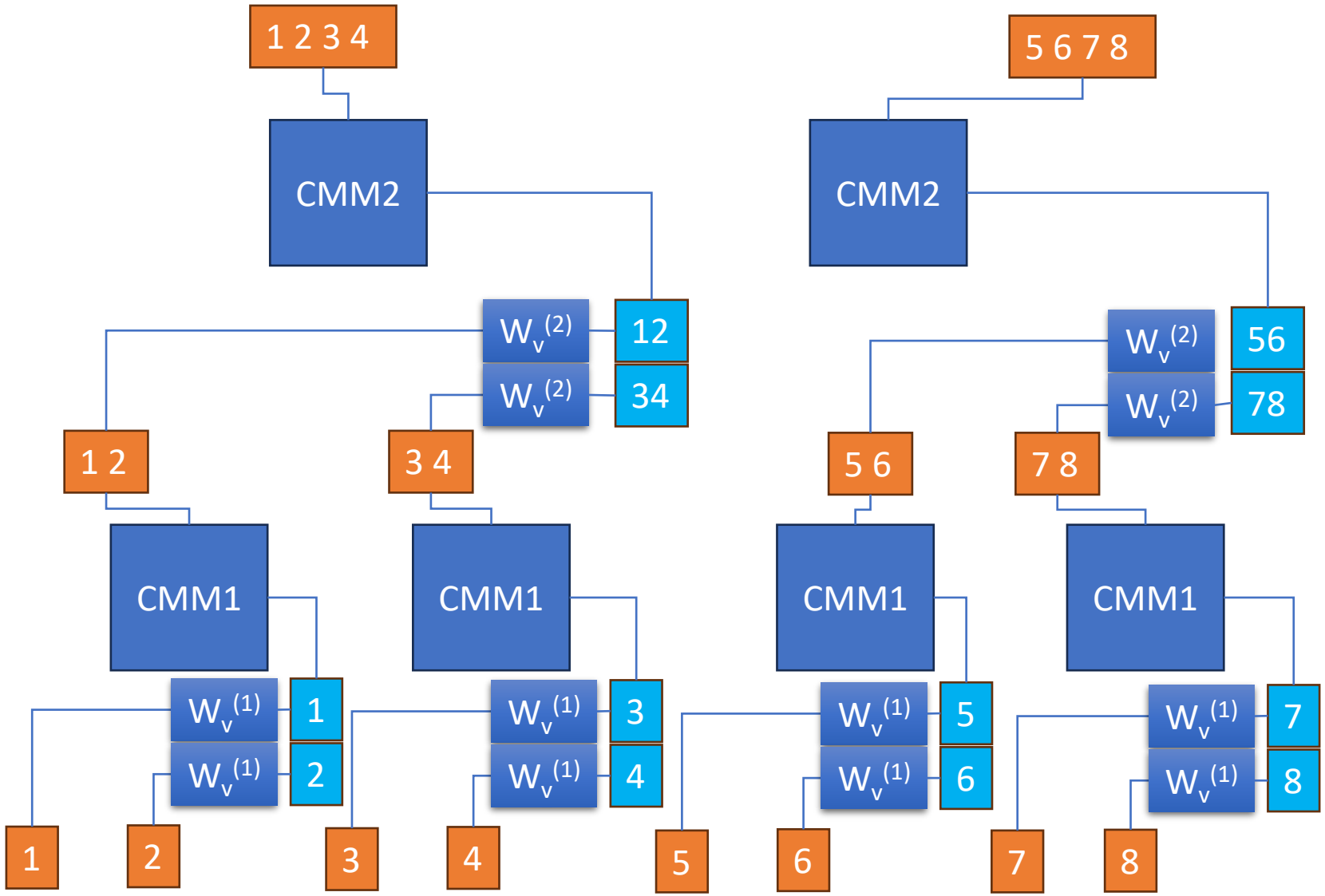

3 4
5 6
7 8
1
1 2
3
3 4
5
5 6
7
7 8
2
4
6
8
$W_v^{(1)}$
1
$W_v^{(1)}$
2
$W_v^{(1)}$
3
$W_v^{(1)}$
4
$W_v^{(1)}$
5
$W_v^{(1)}$
6
$W_v^{(1)}$
7
$W_v^{(1)}$
8
1
2
3
4
5
6
7
8

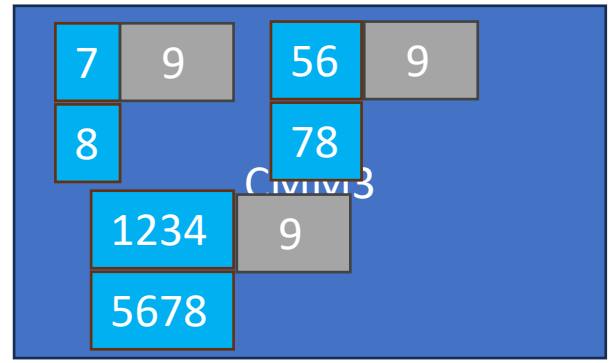

7
9
56
9
8
78
1234
9
5678

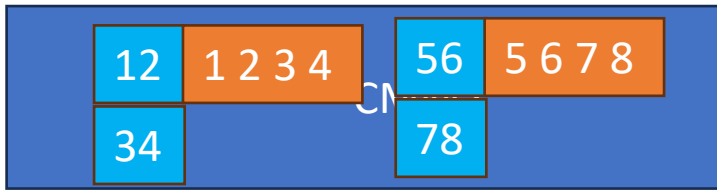

12
1 2 3 4
56
5 6 7 8
34
78

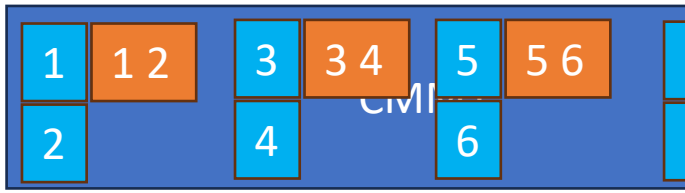

1 2
3 4
5 6
7 8
CMM1
CMM1
CMM1
CMM1
$W_v^{(1)}$
$W_v^{(1)}$
$W_v^{(1)}$
$W_v^{(1)}$
$W_v^{(1)}$
$W_v^{(1)}$
$W_v^{(1)}$
$W_v^{(1)}$
1
2
3
4
5
6
7
8
1
2
3
4
5
6
7
8

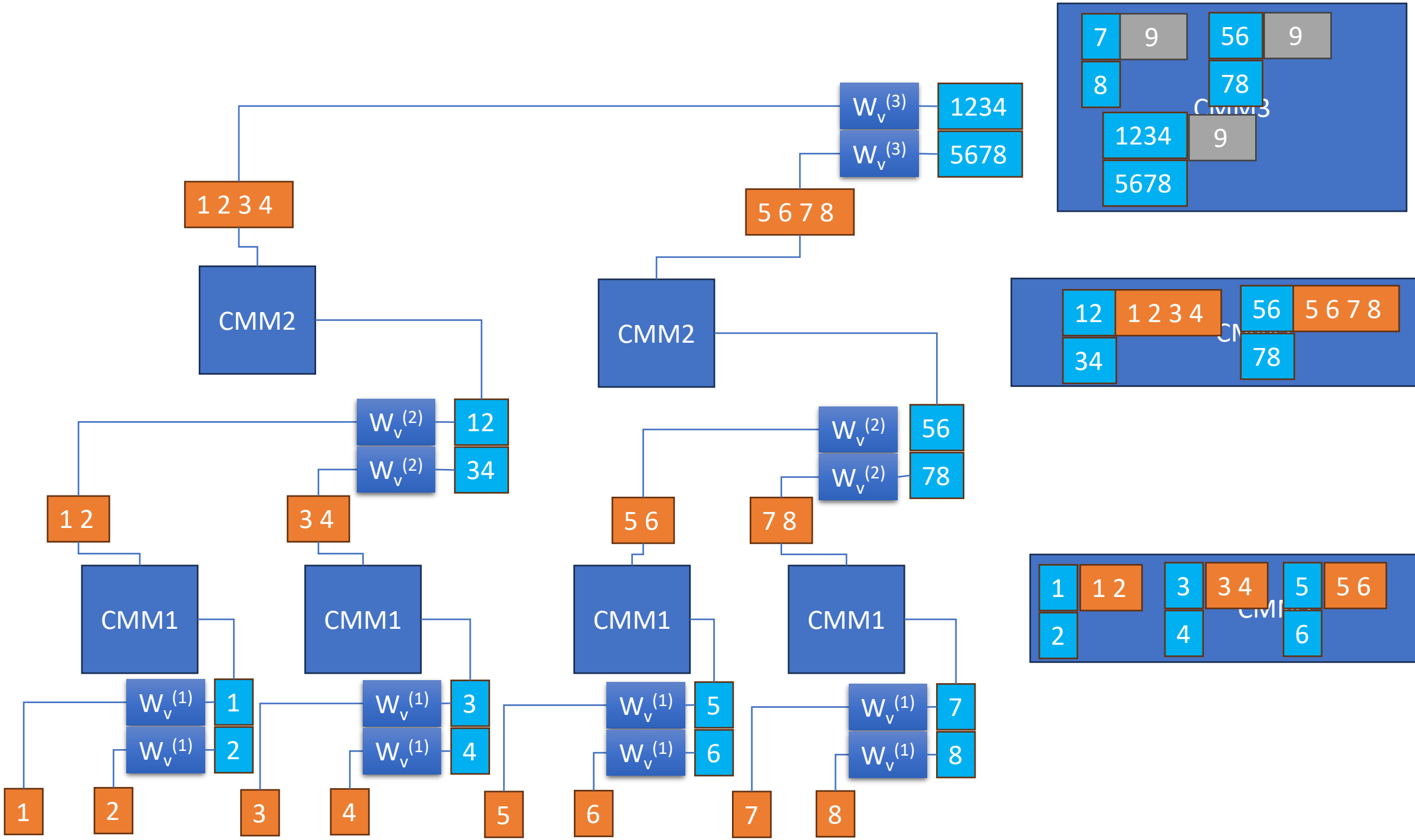

1
1 2
2
3
3 4
4
5
5 6
6

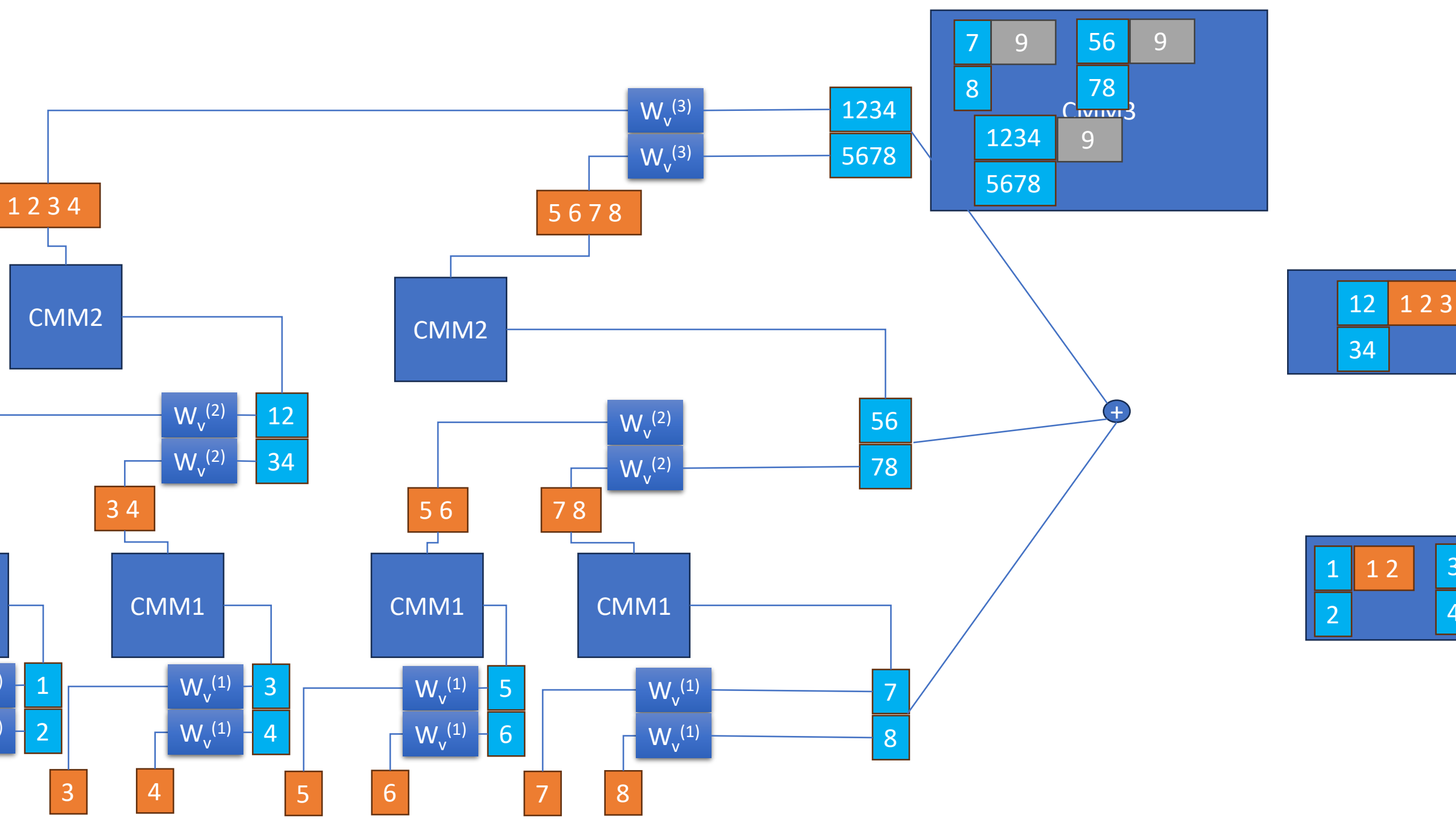

1 2 3 4
CMM2
$W_v^{(2)}$
$W_v^{(2)}$
12
34
1 2
3 4
CMM1
CMM1
$W_v^{(1)}$
$W_v^{(1)}$
1
2
$W_v^{(1)}$
$W_v^{(1)}$
3
4
1
2
3
4
5 6 7 8
CMM2
$W_v^{(2)}$
$W_v^{(2)}$
56
78
5 6
7 8
CMM1
CMM1
$W_v^{(1)}$
$W_v^{(1)}$
5
6
$W_v^{(1)}$
$W_v^{(1)}$
7
8
5
6
7
8

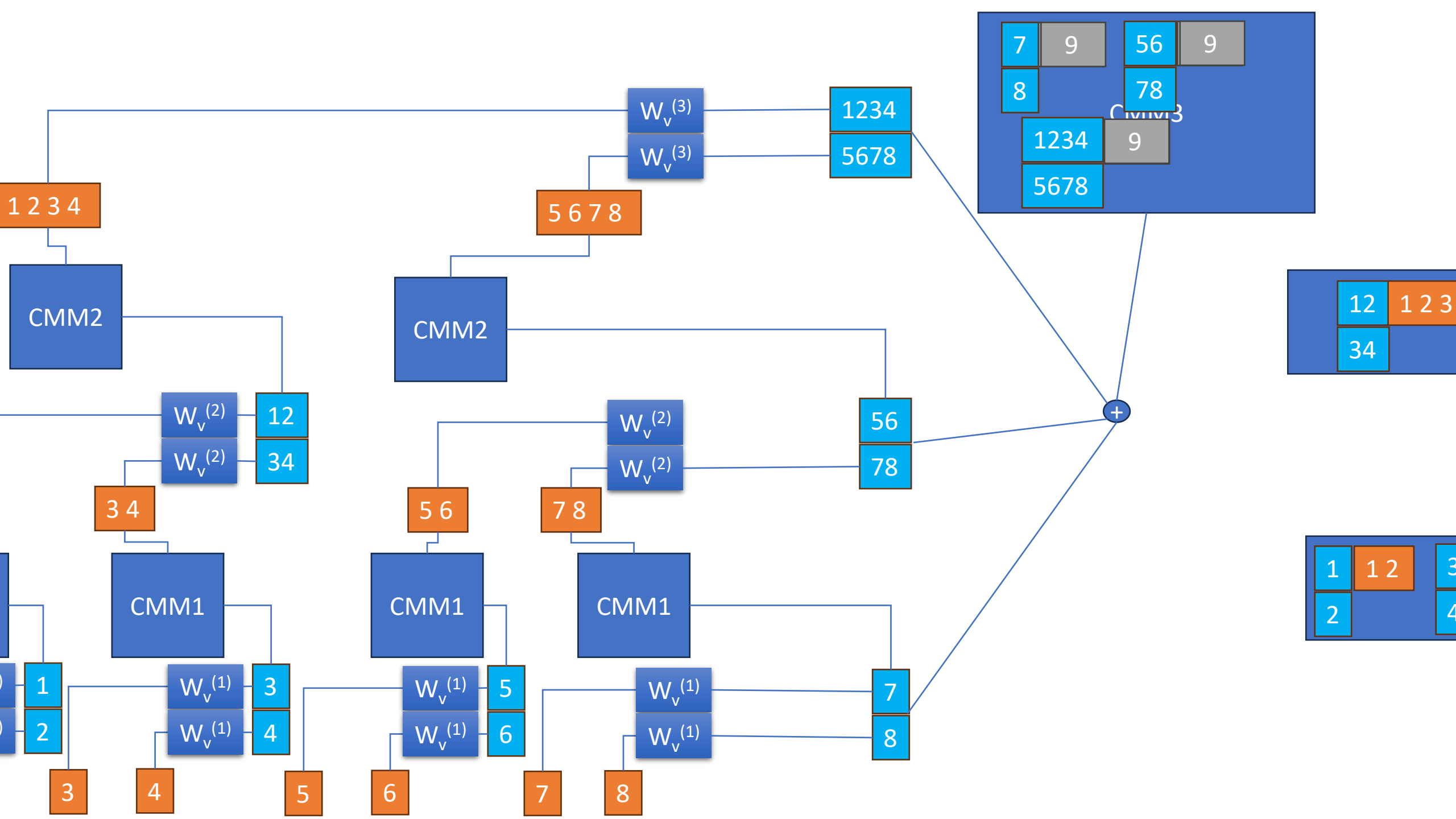

7
9
56
9
8
78
1234
9
5678

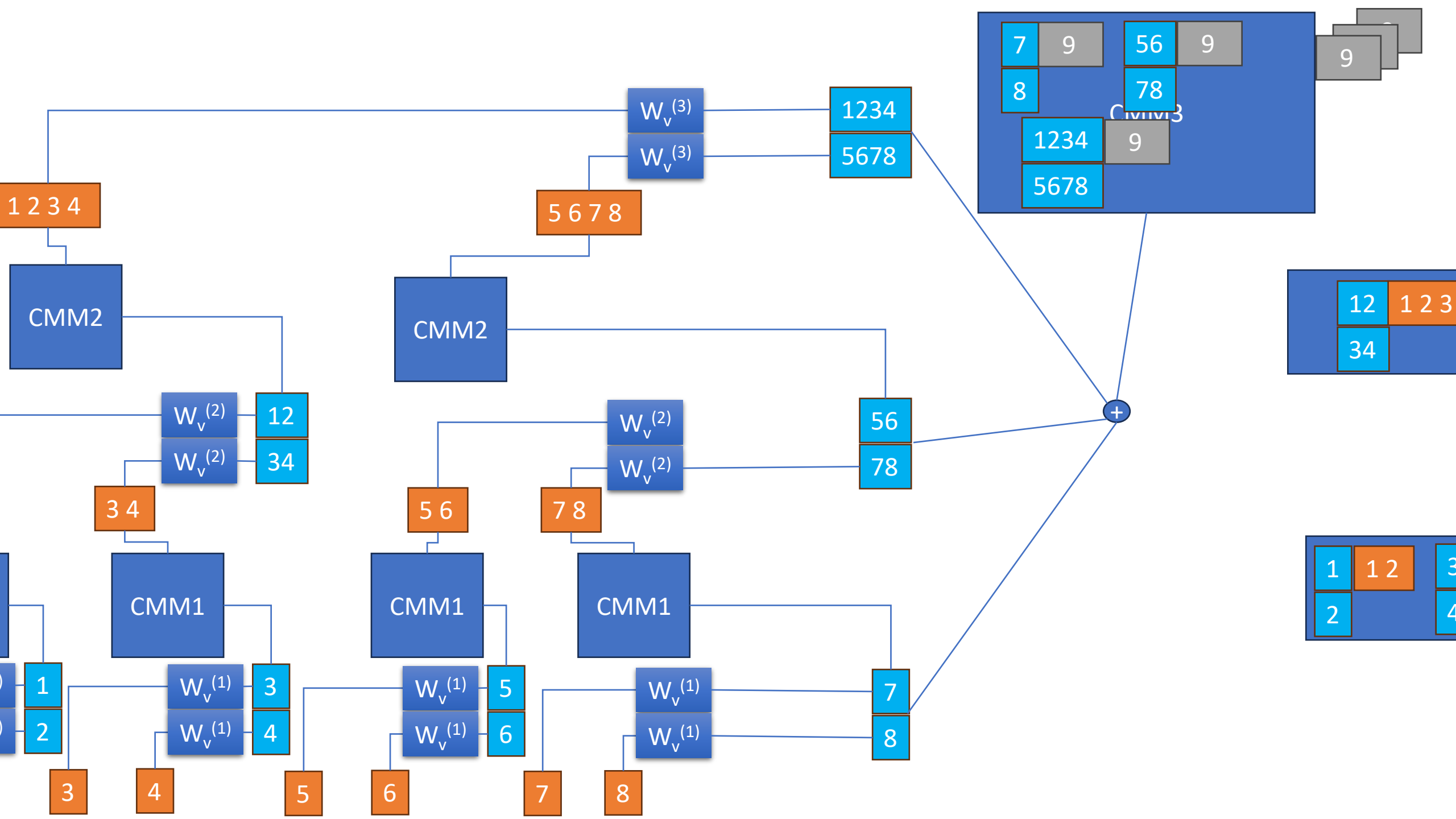

12
1 2 3 4
56
5 6 7 8
34
78


1
1 2
3
3 4
5
5 6
2
4
6


$W_v^{(3)}$
1234
$W_v^{(3)}$
5678
1 2 3 4
5 6 7 8
CMM2
CMM2
$W_v^{(2)}$
12
$W_v^{(2)}$
34
$W_v^{(2)}$
56
$W_v^{(2)}$
78
1 2
3 4
5 6
7 8
CMM1
CMM1
CMM1
CMM1
$W_v^{(1)}$
1
$W_v^{(1)}$
2
$W_v^{(1)}$
3
$W_v^{(1)}$
4
$W_v^{(1)}$
5
$W_v^{(1)}$
6
$W_v^{(1)}$
7
$W_v^{(1)}$
8
1
2
3
4
5
6
7
8
7
9
8
56
9
78
1234
9
5678
12
1 2 3 4
34
56
5 6 7 8
78
1
1 2
2
3
3 4
4
5
5 6
6


1 2 3 4
CMM2
$W_v^{(3)}$
$W_v^{(3)}$
5 6 7 8
CMM2
1234
5678
7
9
56
9
8
78
1234
9
5678
12
1 2 3
34
$W_v^{(2)}$
$W_v^{(2)}$
12
34
3 4
CMM1
$W_v^{(2)}$
$W_v^{(2)}$
56
78
5 6
CMM1
7 8
CMM1
1
2
$W_v^{(1)}$
$W_v^{(1)}$
3
4
$W_v^{(1)}$
$W_v^{(1)}$
5
6
$W_v^{(1)}$
$W_v^{(1)}$
7
8
1
1 2
2
3
4
5
6
7
8


7
9
56
9
8
78
1234
9
5678
CMM3
1234
5678
$W_v^{(3)}$
$W_v^{(3)}$
1 2 3 4
5 6 7 8
CMM2
CMM2
12
1 2 3
34
$W_v^{(2)}$
$W_v^{(2)}$
12
$W_v^{(2)}$
$W_v^{(2)}$
56
78
3 4
5 6
7 8
CMM1
CMM1
CMM1
1
1 2
2
$W_v^{(1)}$
$W_v^{(1)}$
3
4
$W_v^{(1)}$
$W_v^{(1)}$
5
6
$W_v^{(1)}$
$W_v^{(1)}$
7
8
3
4
5
6
7
8


1 2 3 4
CMM2
$W_v^{(3)}$
$W_v^{(3)}$
5 6 7 8
1234
5678
CMM2
$W_v^{(2)}$
$W_v^{(2)}$
12
34
3 4
CMM1
$W_v^{(2)}$
$W_v^{(2)}$
56
78
5 6
7 8
CMM1
CMM1
$W_v^{(1)}$
$W_v^{(1)}$
3
4
$W_v^{(1)}$
$W_v^{(1)}$
5
6
$W_v^{(1)}$
$W_v^{(1)}$
7
8
1
2
3
4
5
6
7
8
7 9
8
56 9
78
1234 9
5678
9
12 1 2 3
34
1 1 2
2
+

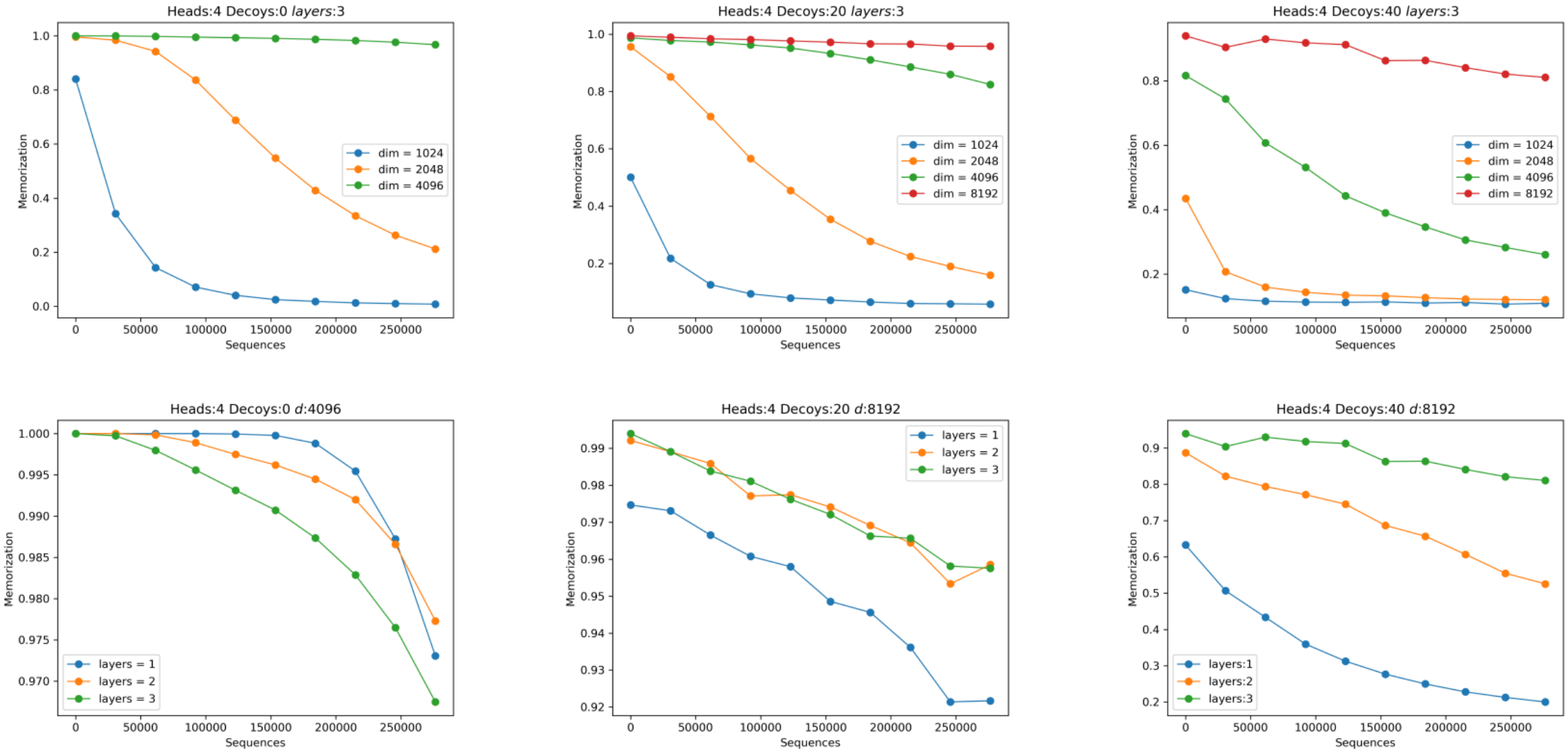
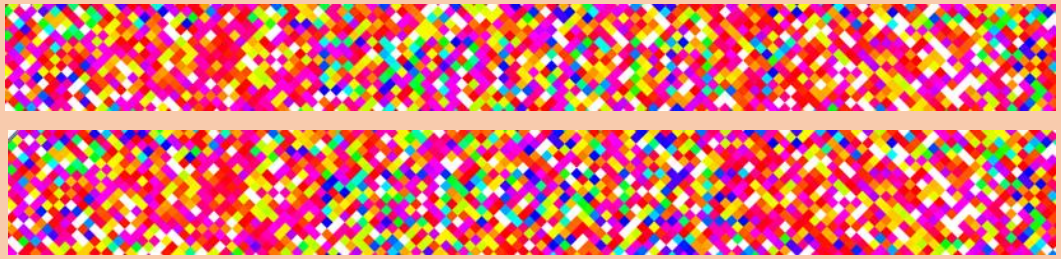


Figure 4: Memorization capacity of MeMo: storing ability with respect to the number of stored sequences. Experiments with increasing complexity of the datasets (number of decoys) and increasing number of layers

The MeMo Approach

How to build a Language Model

CMM =





O =



Title6: Language Modeling with Associative Memory Mechanisms

Fabio Massimo Zanzotto[1,3] **Elena Sofia Ruzzetti**[1] **Giancarlo A. Xompero**[3,1]
Leonardo Ranaldi[2,1] **Davide Venditti**[1] **Federico Ranaldi**[1]
Cristina Giannone[3] **Andrea Favalli**[3] **Raniero Romagnoli**[3]

[1]Human-centric ART, University of Rome Tor Vergata, Italy
[2]University of Edinburgh, Scotland, UK
[3]Almawave S.p.A., Via di Casal Boccone, 188-190 00137, Rome, I

bio o.za @un it



ACCEPTED ACL 2025 VIENNA Findings Papers

Abstract

Memorization is a fundamental ability of Transformer-based Large Language Models, achieved through learning. In this paper, we propose a paradigm shift by designing an architecture to memorize text directly, bearing in mind the principle that memorization pro-

the boundary between generalization and memorization, providing an estimation on their storage capacity. Memorization is not inherently a drawback in language models because it plays a crucial role in handling factual knowledge, which is important for question answering, summarization, or information retrieval. This kind of factual recall

If data are important…









If data are important…

Human-in-the-loop Artificial Intelligence

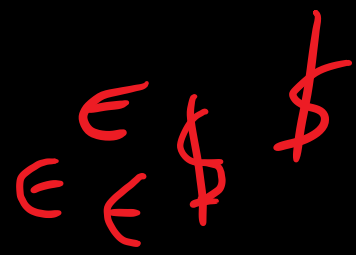
Key Principles

- Any AI system exploits human knowledge
- Hence

any decision of an AI system generating revenues

has to ***repay***

the legitimate owners of this knowledge



Thank you!

What is your question?

In MeMo (Style), we have still a lot to do…

- Invent strategies to learn afterward
- Inject linguistic knowledge: syntax, semantics, pragmatics!
- Study new architectures
- Decide what to store, when, and how
- Study what is missing in Neural Networks
  (for example, variable binding)

This is my question for you





MeMo is a FISA project!



FISA

TOR VERGATA
UNIVERSITY OF ROME

ALMAWAVE