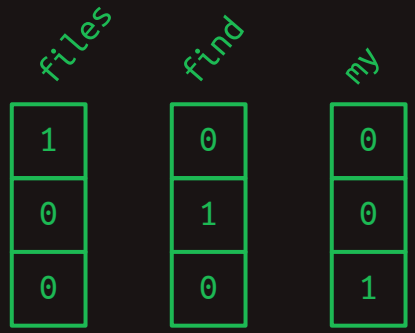
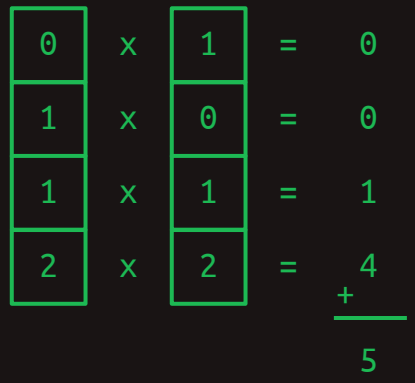
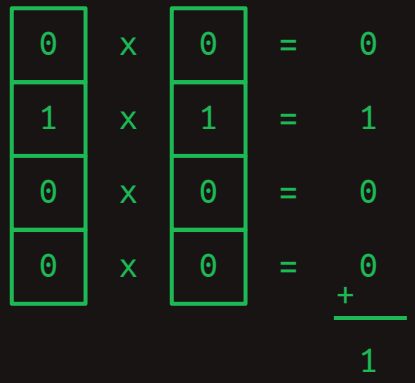
How Transformers Generate and Translate Sequences: The Nuts and Bolts

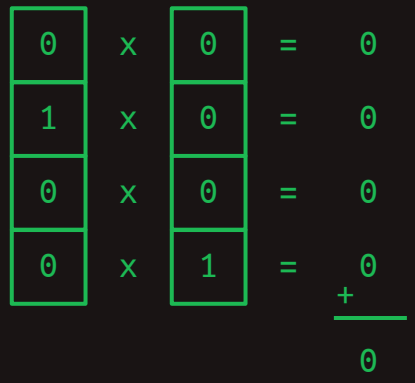
Brandon Rohrer

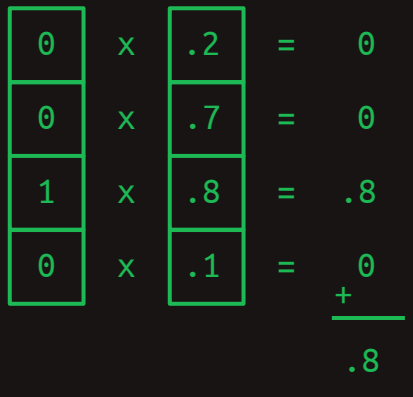

files
find
my
1
0
0
0
1
0
0
0
1

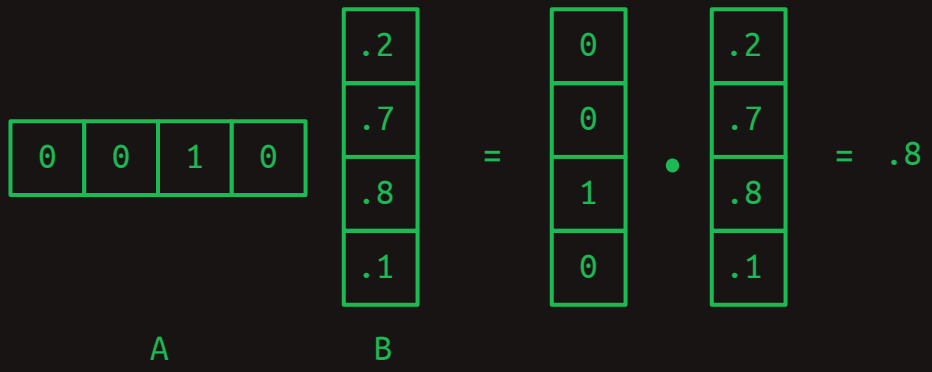

find
my
files
0
0
1
1
0
0
0
1
0

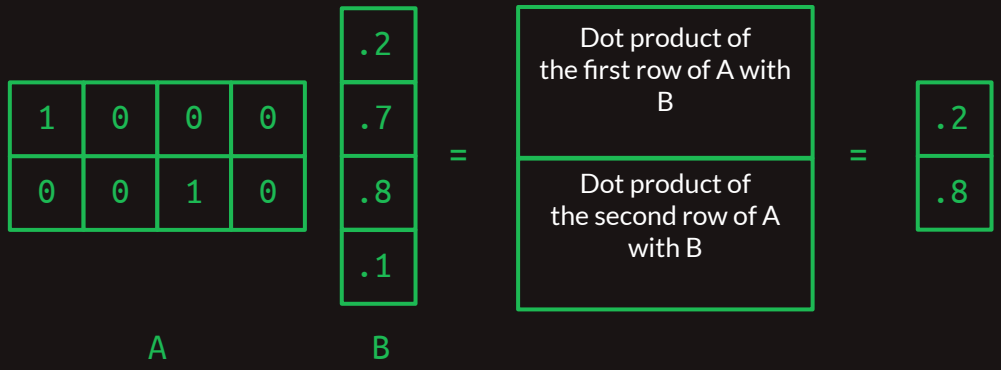

0 x 1 = 0
1 x 0 = 0
1 x 1 = 1
2 x 2 = 4
+
5

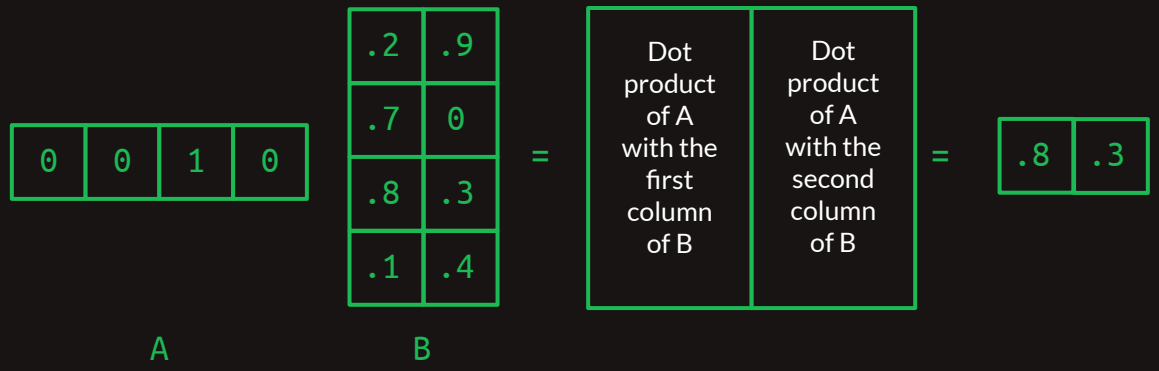

0 x 0 = 0
1 x 1 = 1
0 x 0 = 0
0 x 0 = 0
+
1

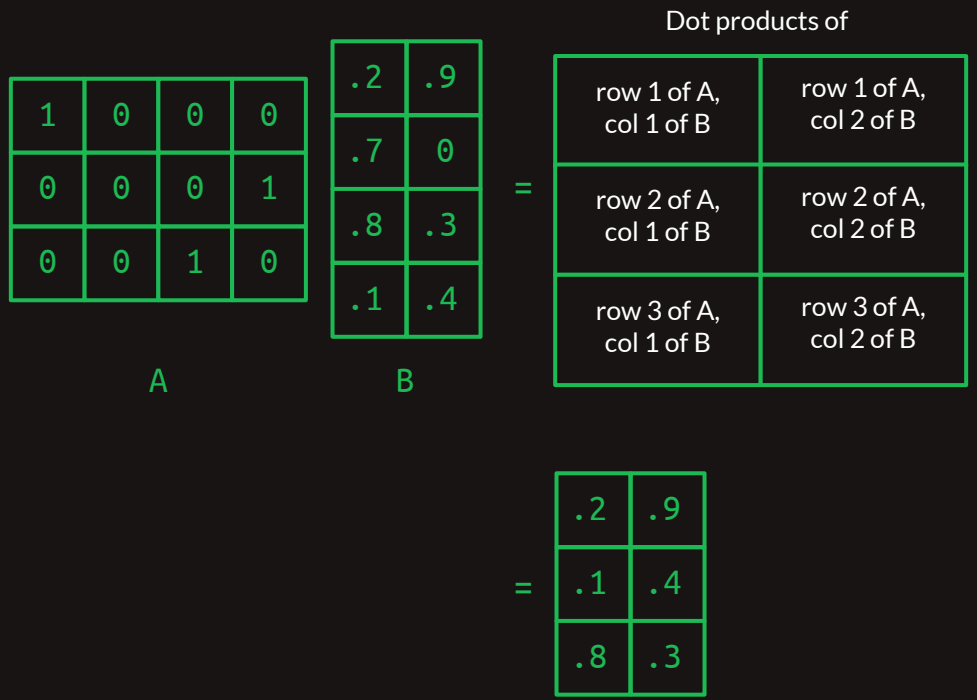

0 x 0 = 0
1 x 0 = 0
0 x 0 = 0
0 x 1 = 0
+
0

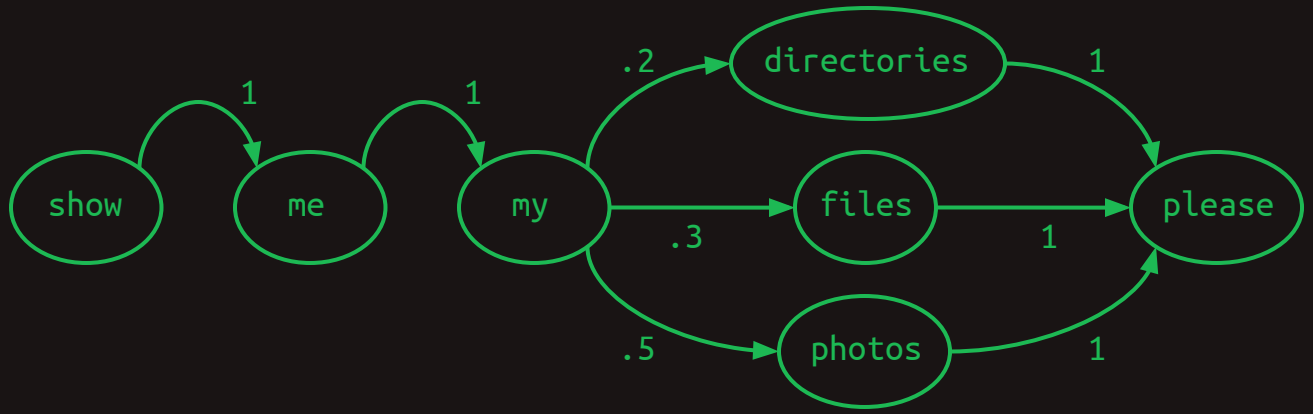

0 x .2 = 0
0 x .7 = 0
1 x .8 = .8
0 x .1 = 0
+
.8

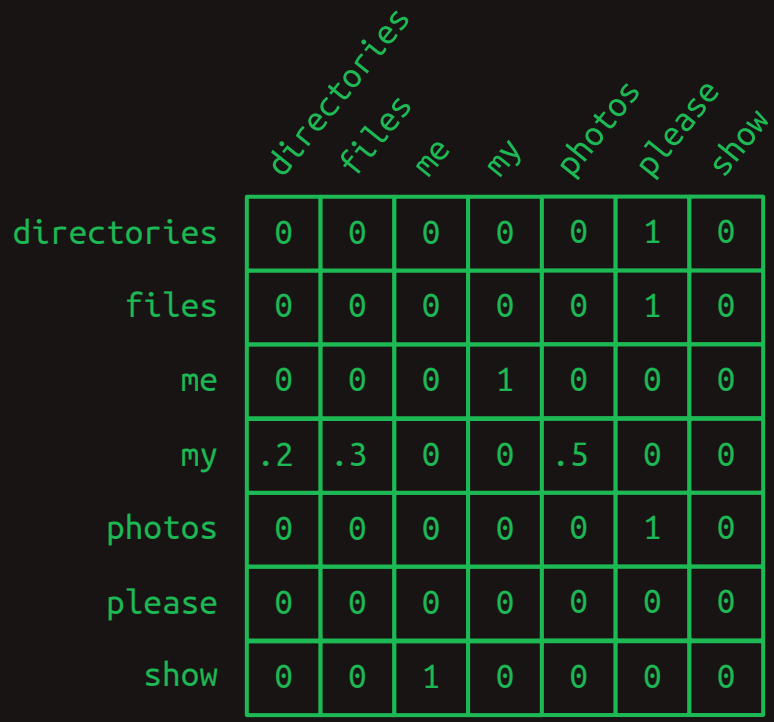

0
0
1
0
A
.2
.7
.8
.1
B
=
0
0
1
0
•
.2
.7
.8
.1
= .8

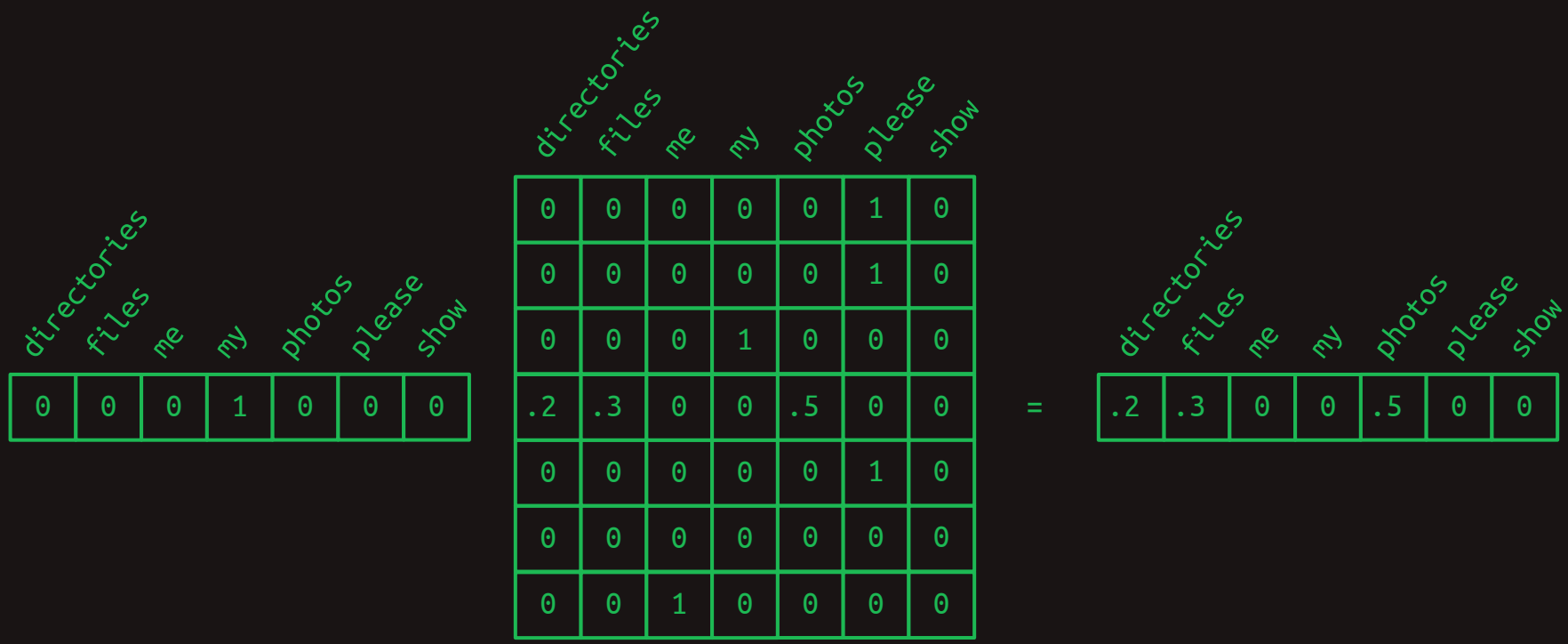

1 0 0 0
0 0 1 0
A
.2
.7
.8
.1
B
=
Dot product of the first row of A with B
Dot product of the second row of A with B
=
.2
.8

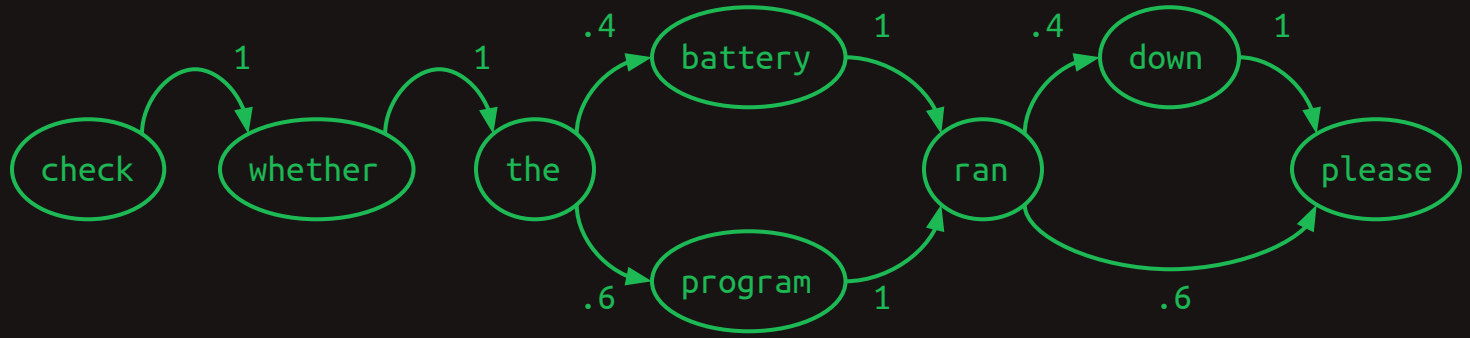

0
0
1
0
.2
.9
.7
0
.8
.3
.1
.4
=
Dot product of A with the first column of B
Dot product of A with the second column of B
=
.8
.3
A
B


Dot products of
1 0 0 0
0 0 0 1
0 0 1 0
A
.2 .9
.7 0
.8 .3
.1 .4
B
=
row 1 of A, col 1 of B
row 1 of A, col 2 of B
row 2 of A, col 1 of B
row 2 of A, col 2 of B
row 3 of A, col 1 of B
row 3 of A, col 2 of B
=
.2 .9
.1 .4
.8 .3


show
me
my
directories
files
photos
please
1
1
.2
.3
.5
1
1
1

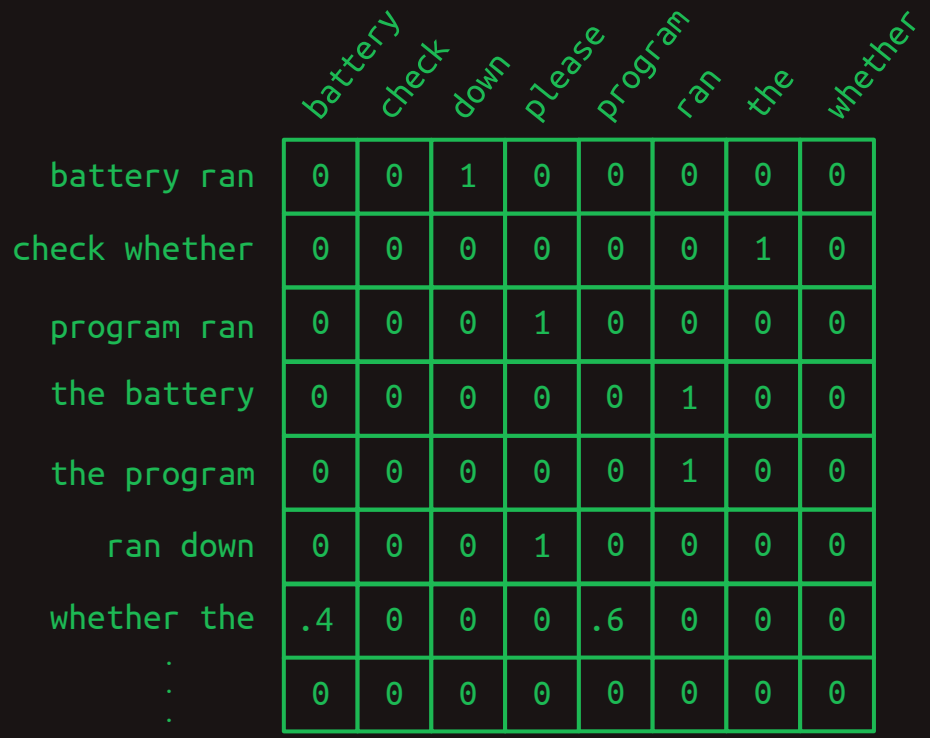

directories
files
me
my
photos
please
show
directories 0 0 0 0 0 1 0
files 0 0 0 0 0 1 0
me 0 0 0 1 0 0 0
my .2 .3 0 0 .5 0 0
photos 0 0 0 0 0 1 0
please 0 0 0 0 0 0 0
show 0 0 1 0 0 0 0

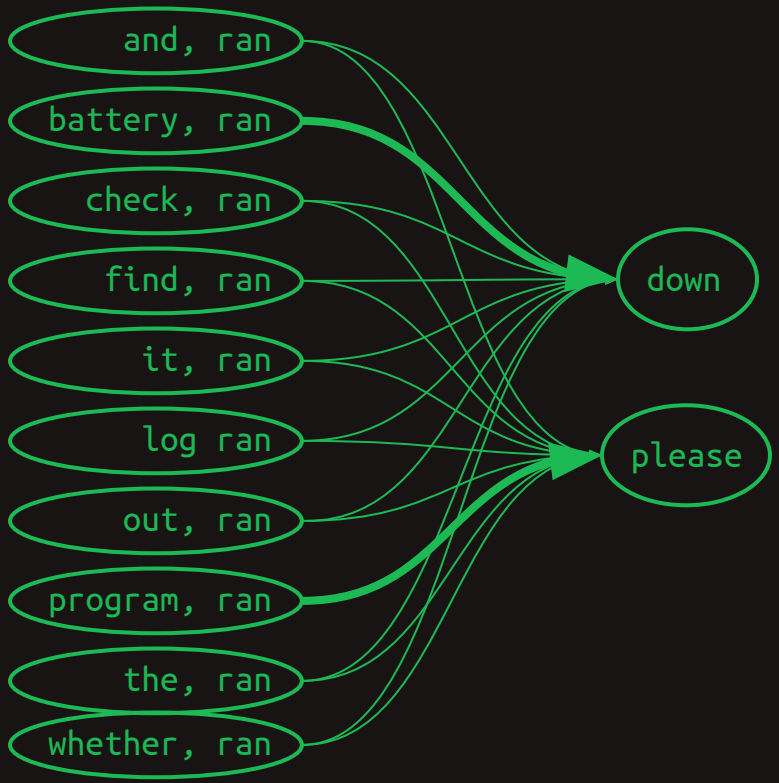

directories
files
me
my
photos
please
show
0 0 0 1 0 0 0
directories
files
me
my
photos
please
show
0 0 0 0 0 1 0
0 0 0 0 0 1 0
0 0 0 1 0 0 0
.2 .3 0 0 .5 0 0
0 0 0 0 0 1 0
0 0 0 0 0 0 0
0 0 1 0 0 0 0
=
directories
files
me
my
photos
please
show
.2 .3 0 0 .5 0 0

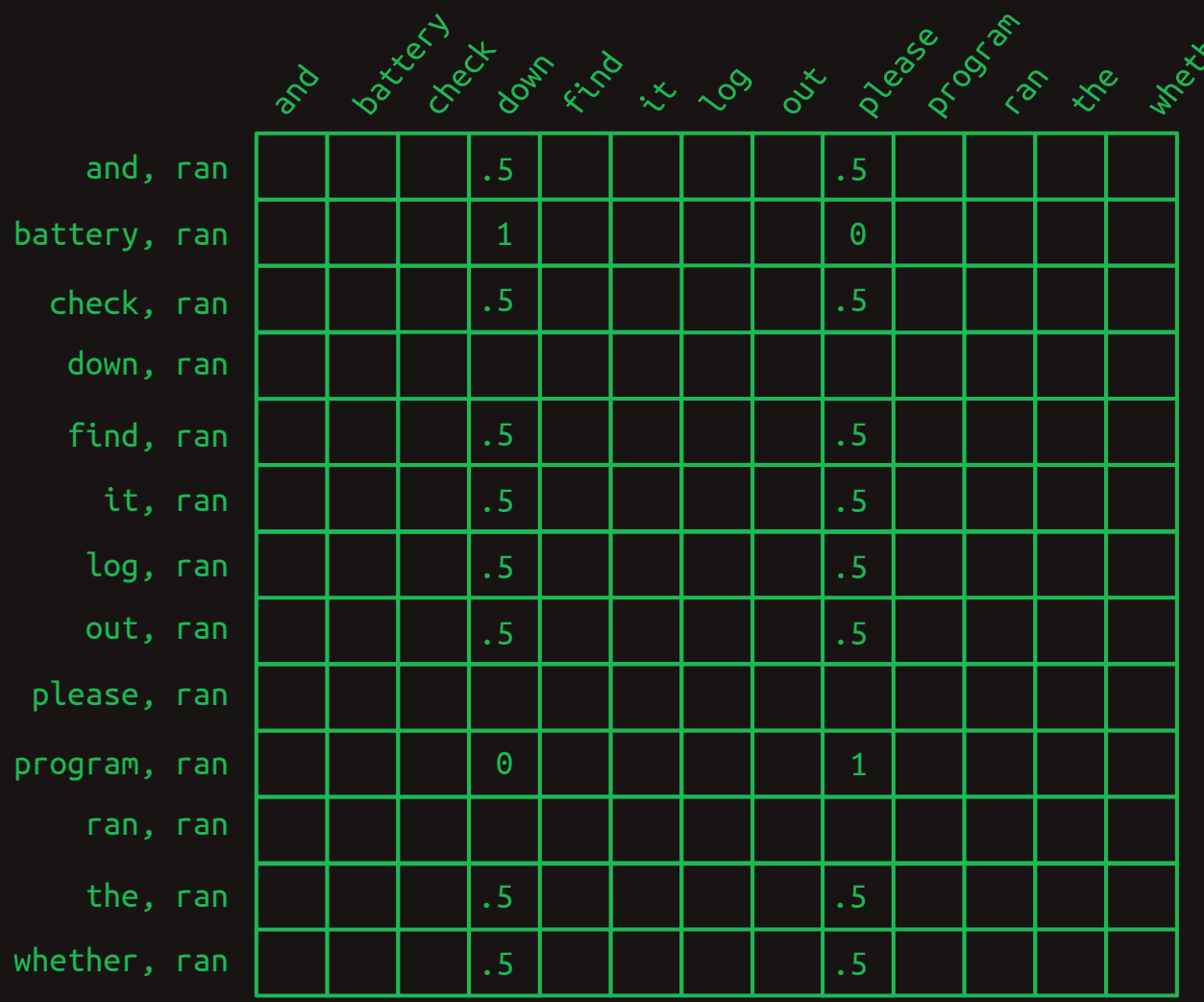

check
whether
the
battery
program
ran
down
please
1
1
.4
1
.6
1
.4
1
.6

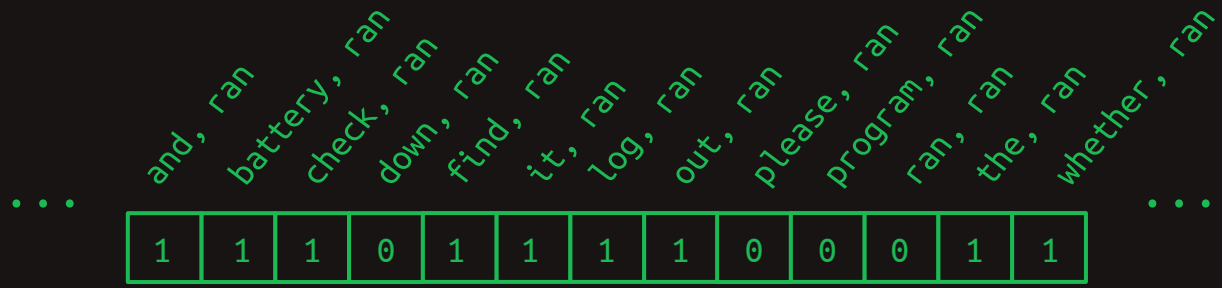
	battery	check	down	please	program	ran	the	whether
battery	0	0	0	0	0	1	0	0
check	0	0	0	0	0	0	0	1
down	0	0	0	1	0	0	0	0
please	0	0	0	0	0	0	0	0
program	0	0	0	0	0	1	0	0
ran	0	0	.4	.6	0	0	0	0
the	.4	0	0	0	.6	0	0	0
whether	0	0	0	0	0	0	1	0

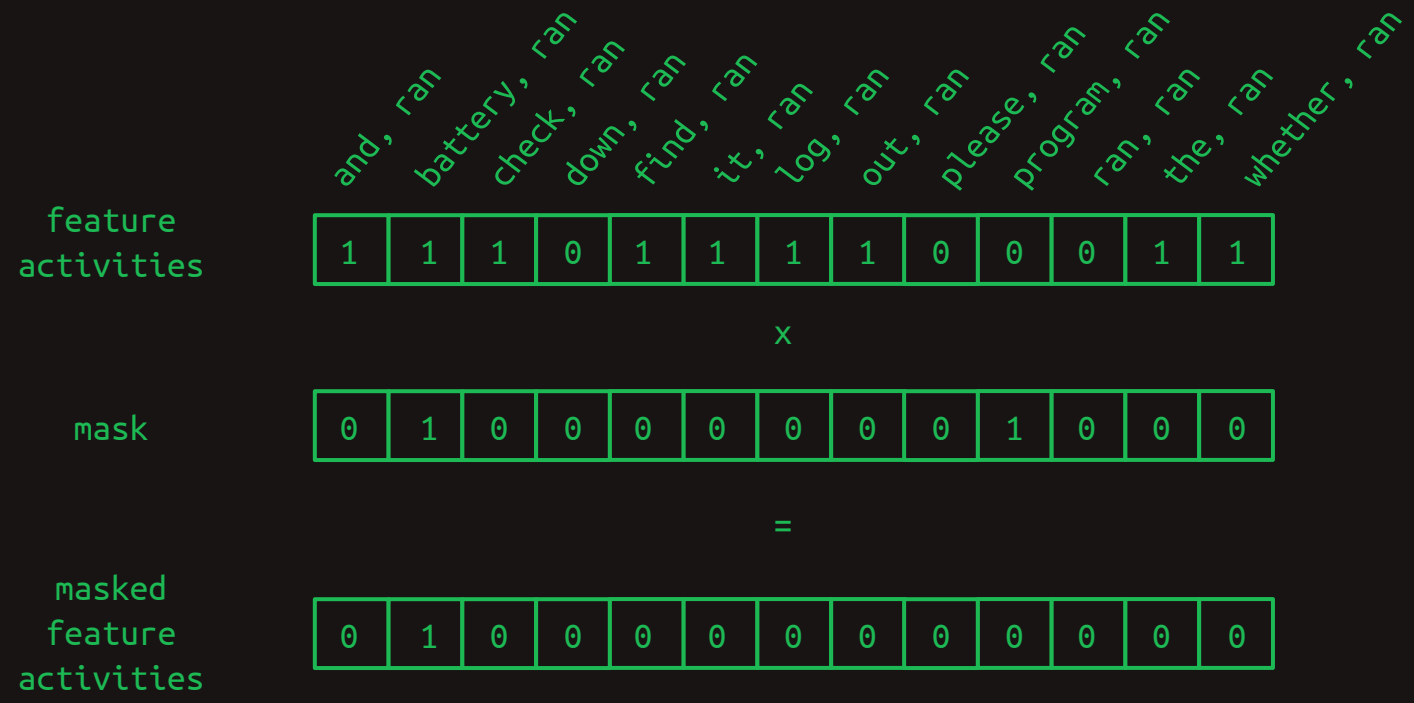

1
battery ran
down
program ran
please
1

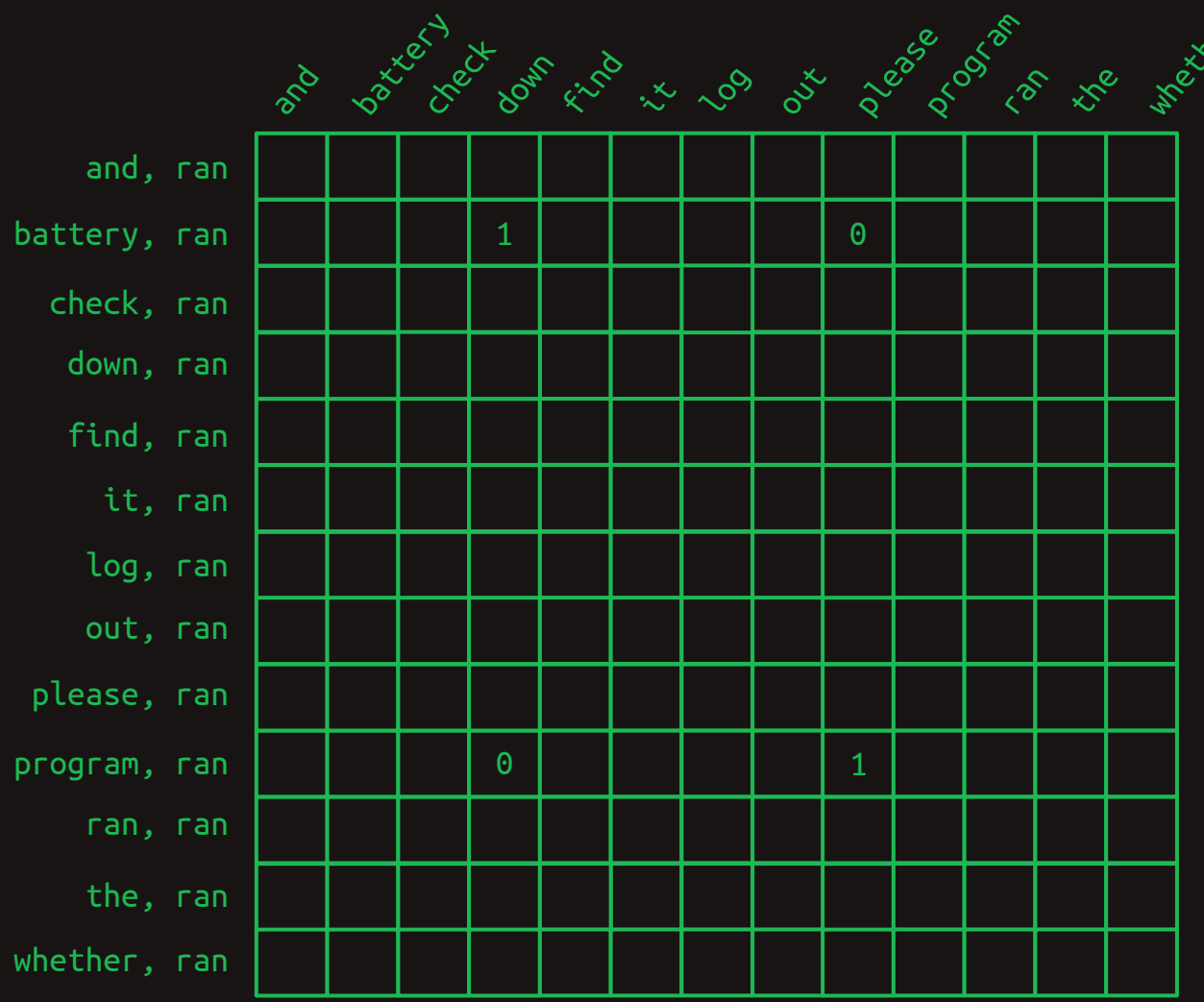

battery
check
down
please
program
ran
the
whether
battery ran 0 0 1 0 0 0 0 0
check whether 0 0 0 0 0 0 1 0
program ran 0 0 0 1 0 0 0 0
the battery 0 0 0 0 0 1 0 0
the program 0 0 0 0 0 1 0 0
ran down 0 0 0 1 0 0 0 0
whether the .4 0 0 0 .6 0 0 0
0 0 0 0 0 0 0 0


and, ran
battery, ran
check, ran
find, ran
it, ran
log ran
out, ran
program, ran
the, ran
whether, ran
down
please

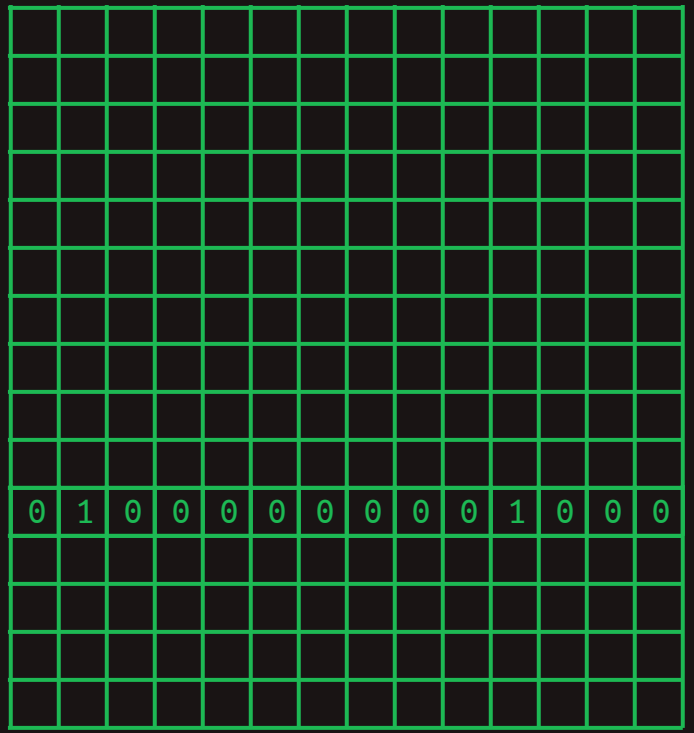

and
battery
check
down
find
it
log
out
please
program
ran
the
whether
and, ran
battery, ran
check, ran
down, ran
find, ran
it, ran
log, ran
out, ran
please, ran
program, ran
ran, ran
the, ran
whether, ran
.5
1
0
.5
.5
.5
.5
.5
0
1
.5
.5


...
and, ran
battery, ran
check, ran
down, ran
find, ran
it, ran
log, ran
out, ran
please, ran
program, ran
ran, ran
the, ran
whether, ran
...
1 1 1 0 1 1 1 1 0 0 0 1 1


and, ran
battery, ran
check, ran
down, ran
find, ran
it, ran
log, ran
out, ran
please, ran
program, ran
ran, ran
the, ran
whether, ran
feature activities
1 1 1 0 1 1 1 1 0 0 0 1 1
x
mask
0 1 0 0 0 0 0 0 0 1 0 0 0
=
masked feature activities
0 1 0 0 0 0 0 0 0 0 0 0 0


and
battery
check
down
find
it
log
out
please
program
ran
the
whether
and, ran
battery, ran
check, ran
down, ran
find, ran
it, ran
log, ran
out, ran
please, ran
program, ran
ran, ran
the, ran
whether, ran
1
0
0
1

all the masks (keys: K^T)



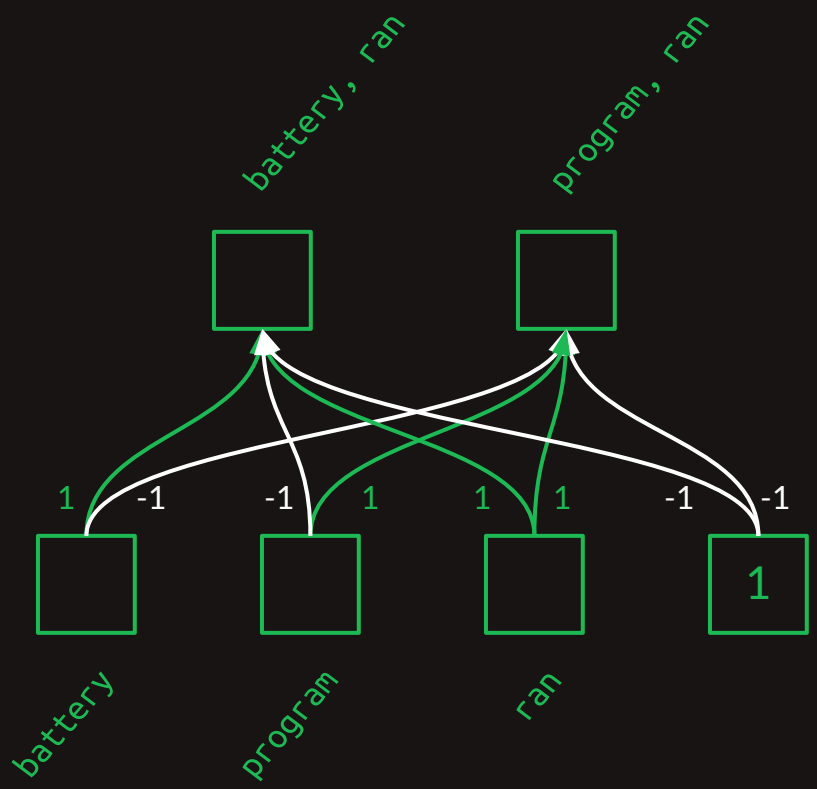
one-hot feature vector (query: Q)

0	0	0	0	0	0	0	0	0	1	0	0	0	0

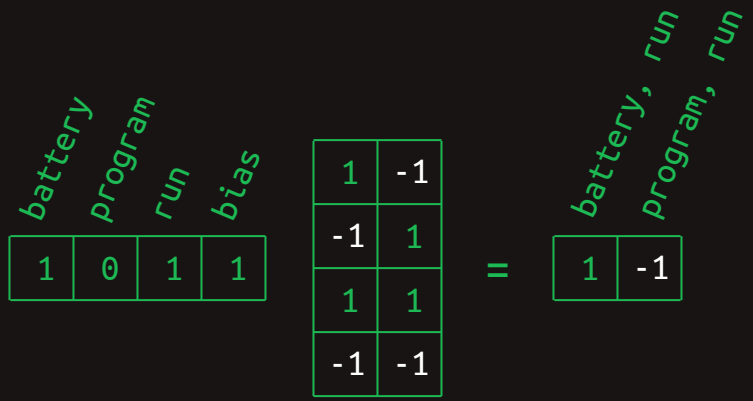
feature-specific mask

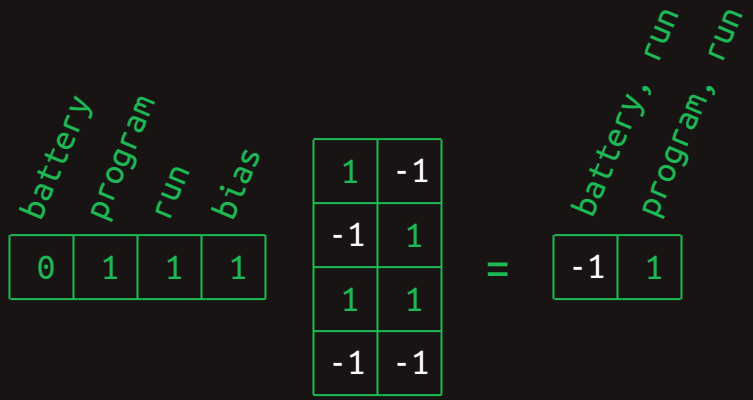
= | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

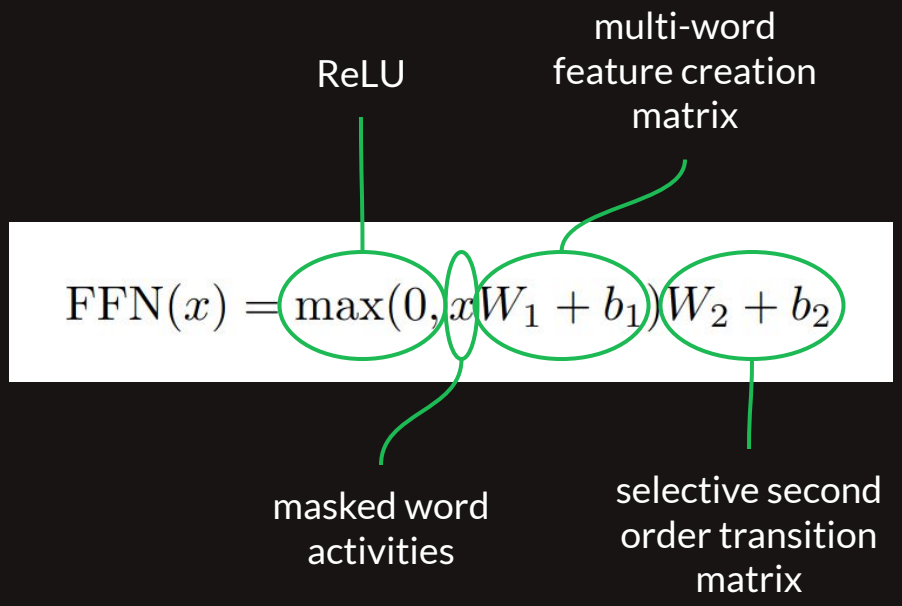
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

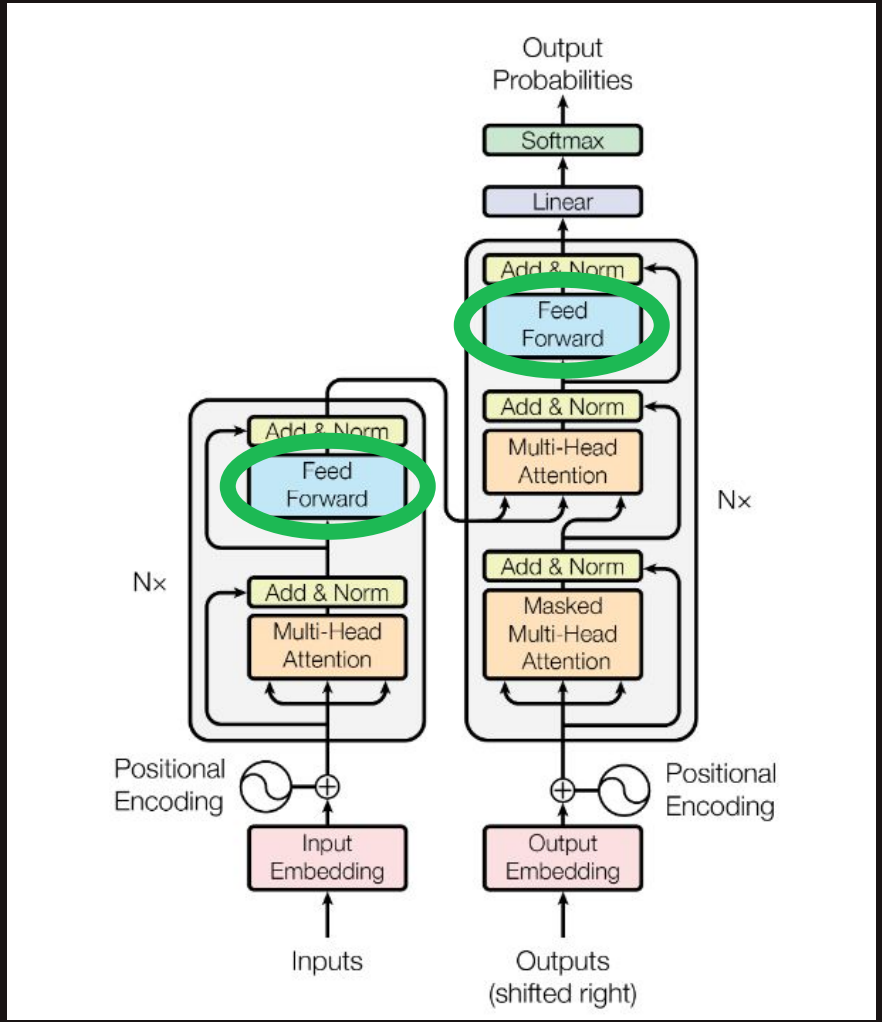


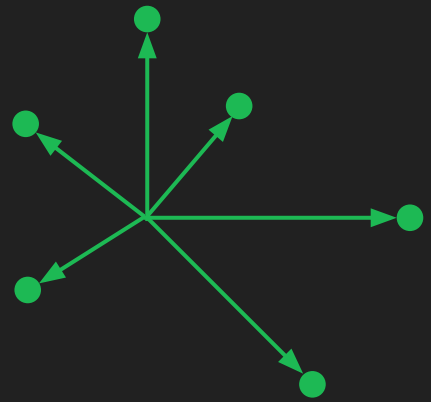
	battery, run	program, run
battery	1	-1
program	-1	1
run	1	1
bias	-1	-1


battery
program
run
bias
1
0
1
1
1
-1
-1
1
1
1
-1
-1
=
battery, run
program, run
1
-1

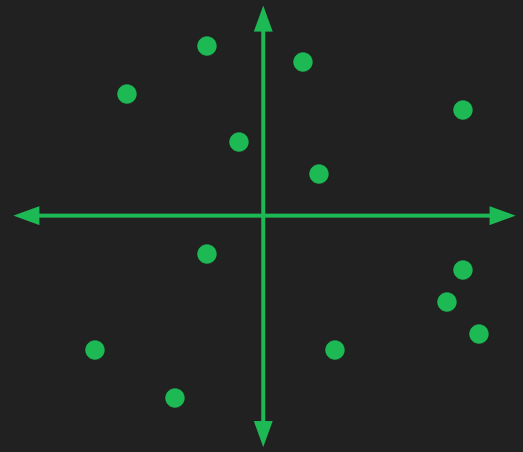

battery
program
run
bias
0
1
1
1
1
-1
-1
1
1
1
-1
-1
=
battery, run
program, run
-1
1

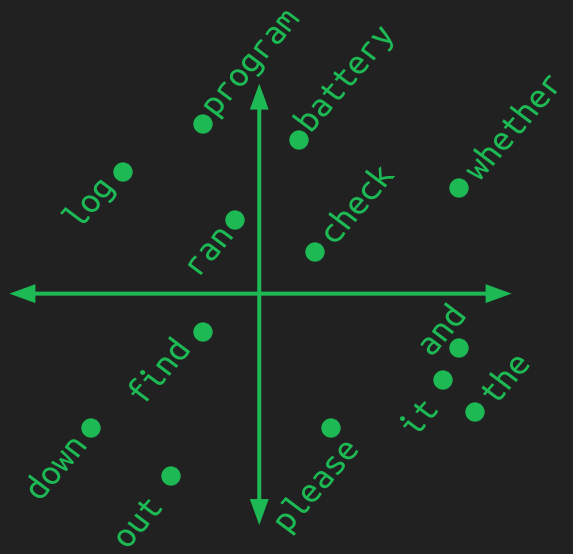

ReLU
multi-word feature creation matrix
$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$
masked word activities
selective second order transition matrix


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)



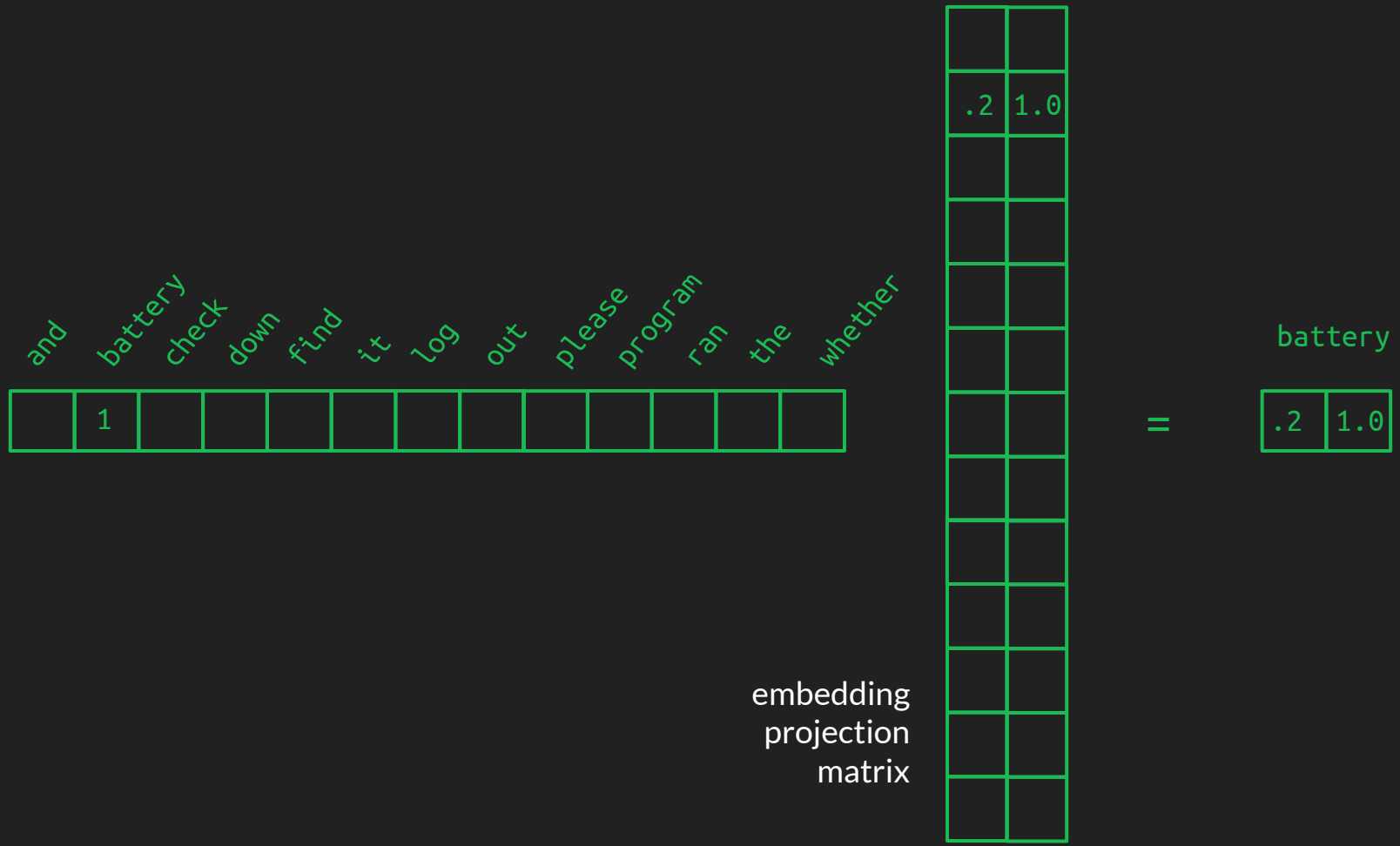
embedding
in 2 dimensions

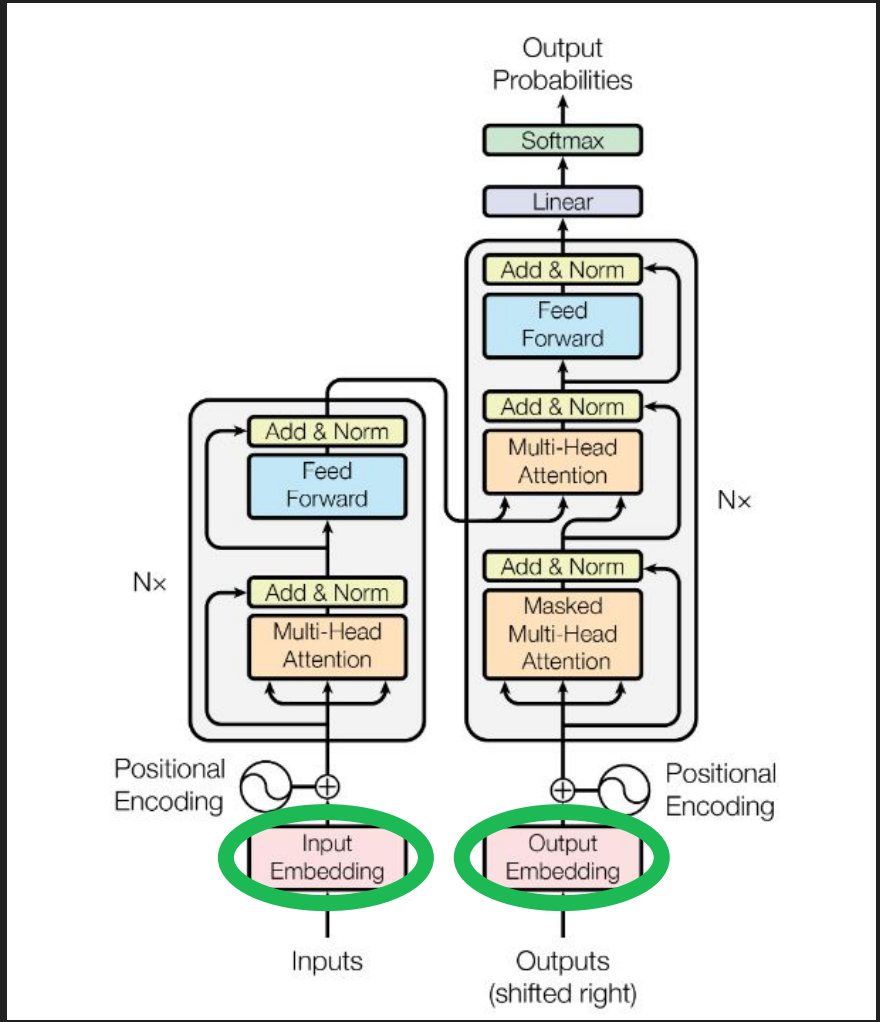


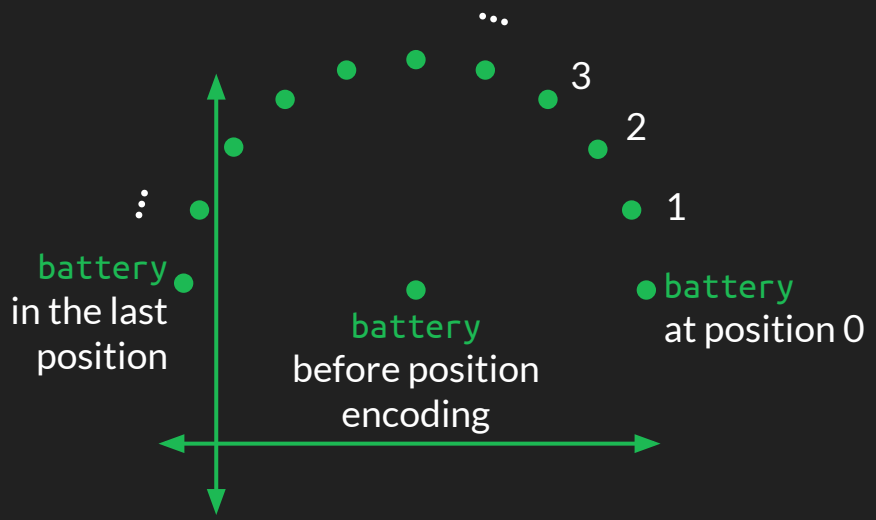


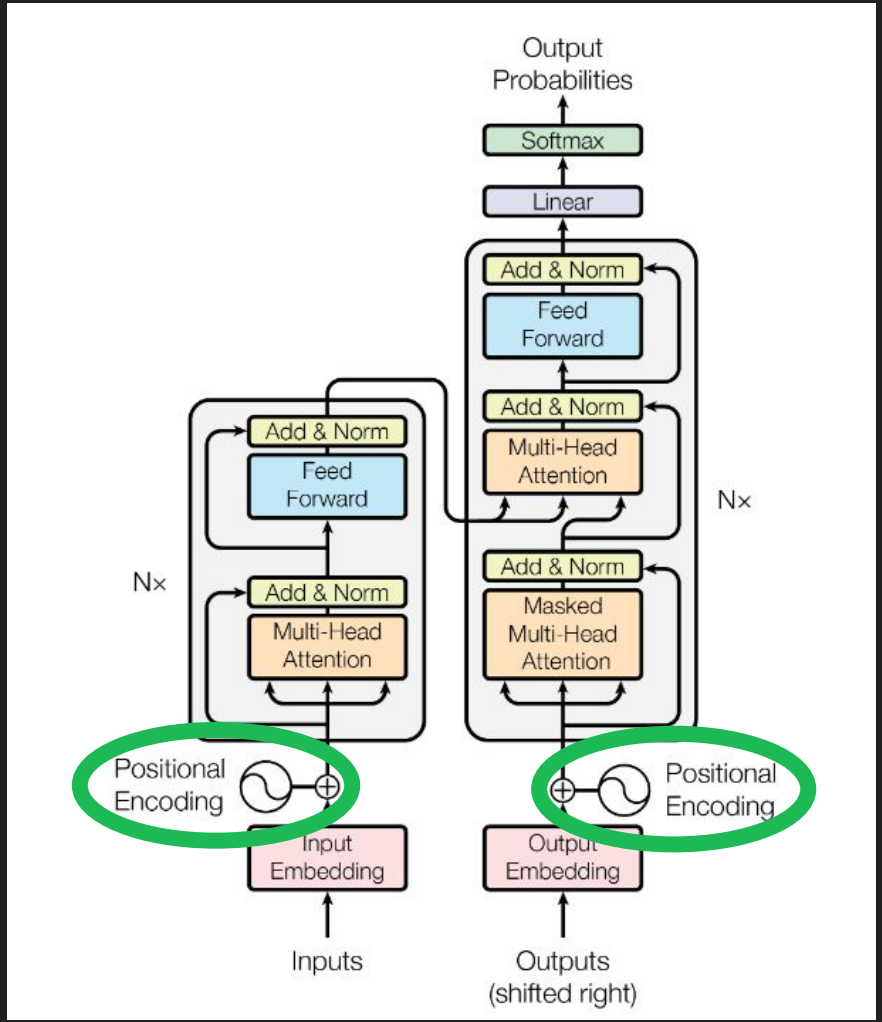
.2	1.0	battery
-.3	1.1	program
-.9	-.8	down
-.5	-1.2	out

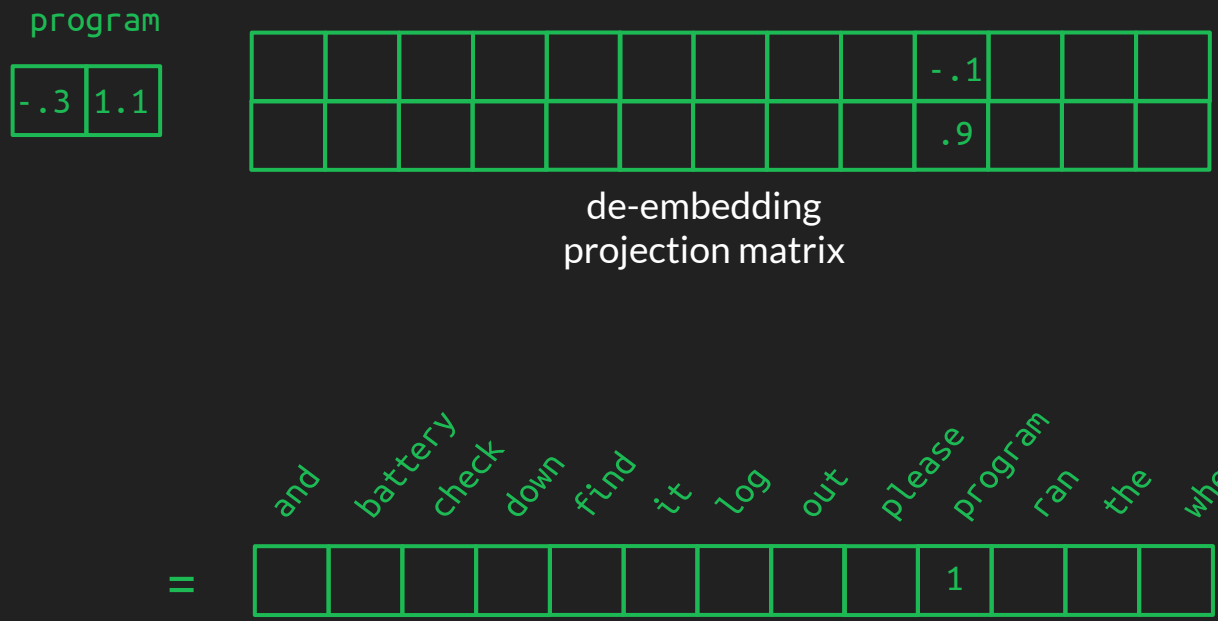
⋮

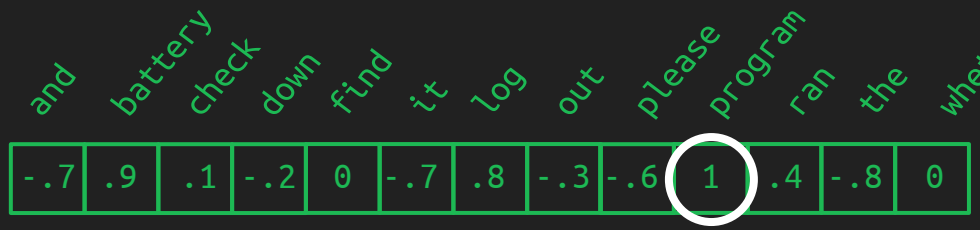

and
battery
check
down
find
it
log
out
please
program
ran
the
whether
1
.2
1.0
embedding
projection
matrix
=
battery
.2
1.0

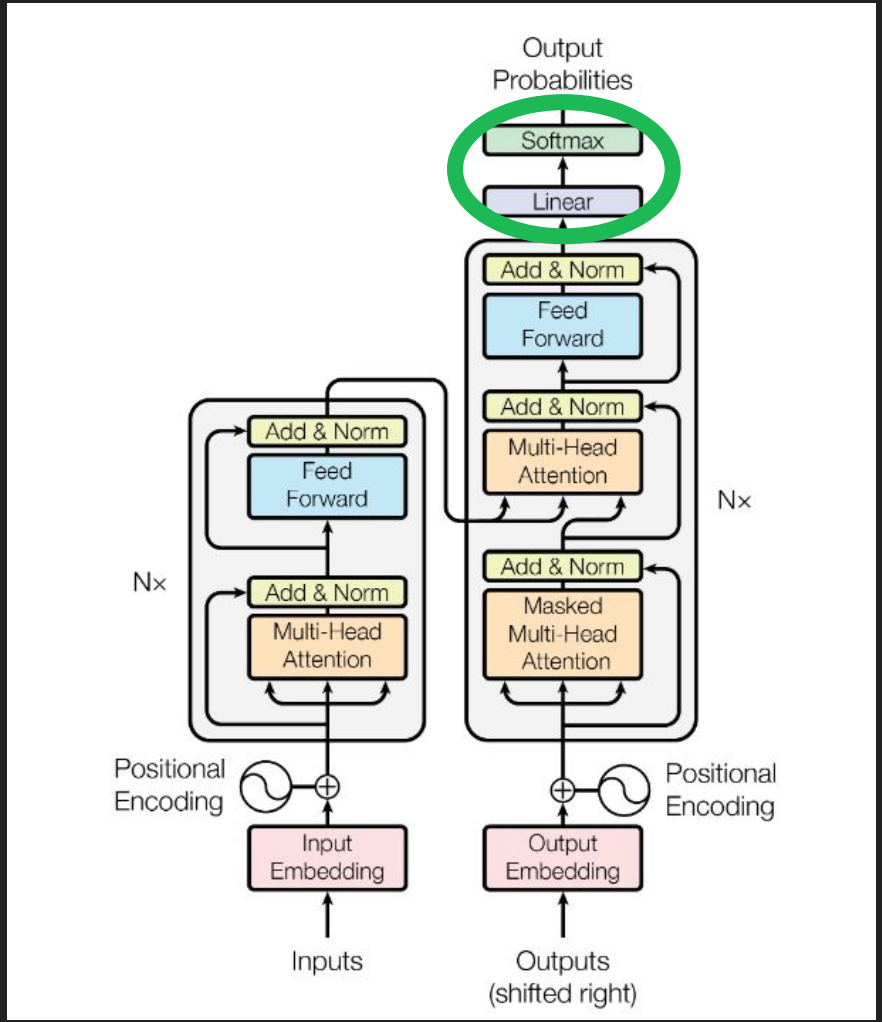

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

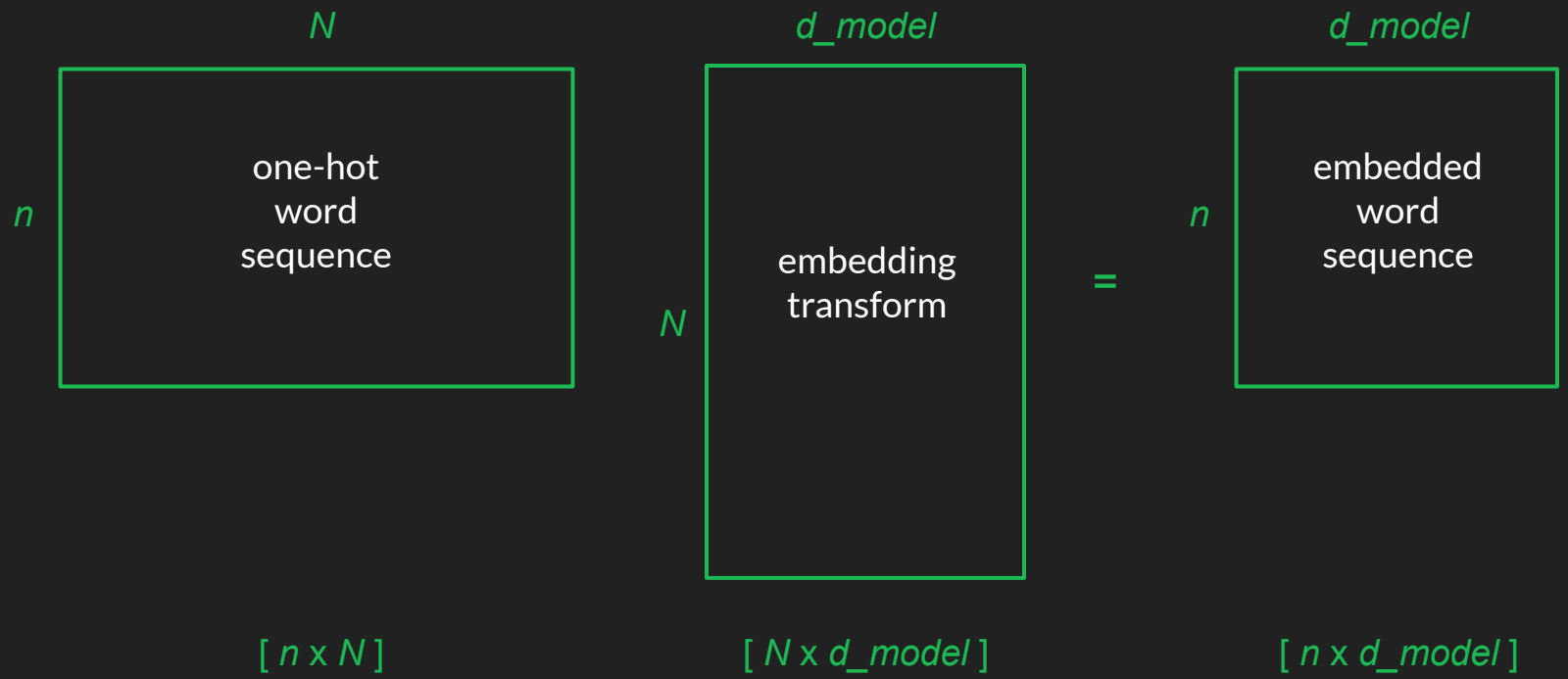

3
2
1
battery
in the last
position
battery
before position
encoding
battery
at position 0

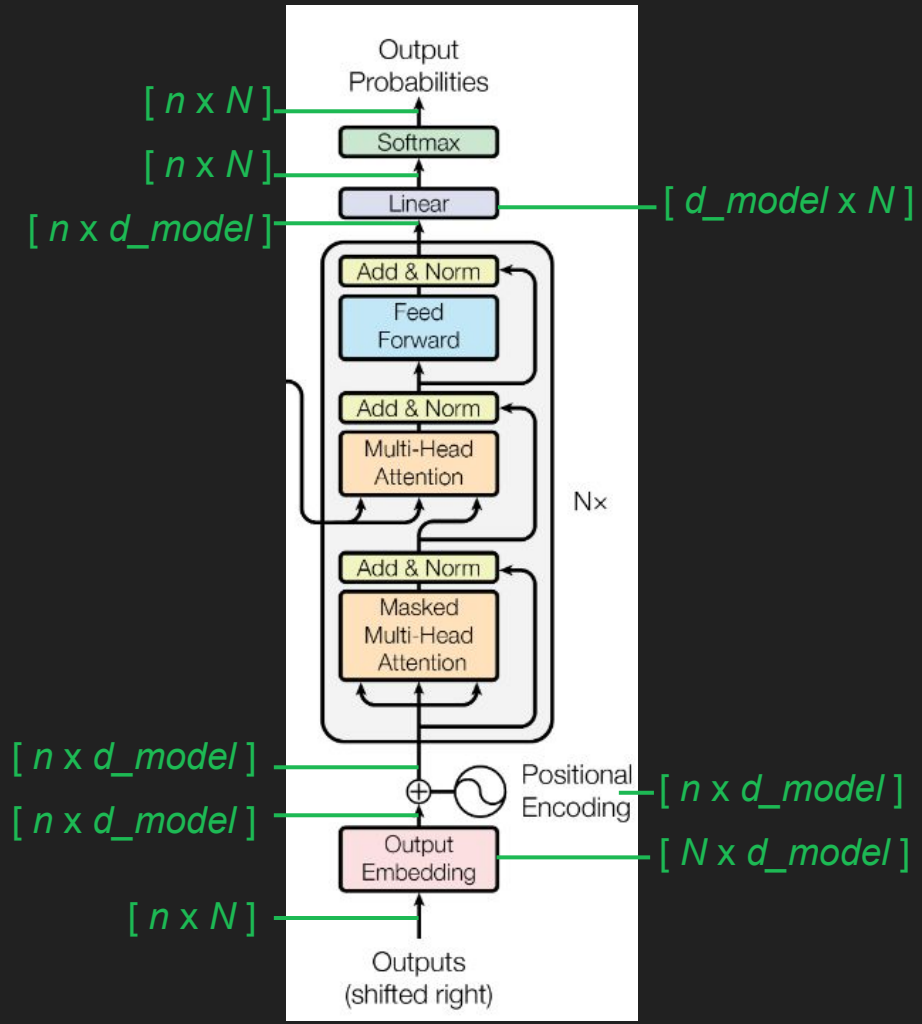

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

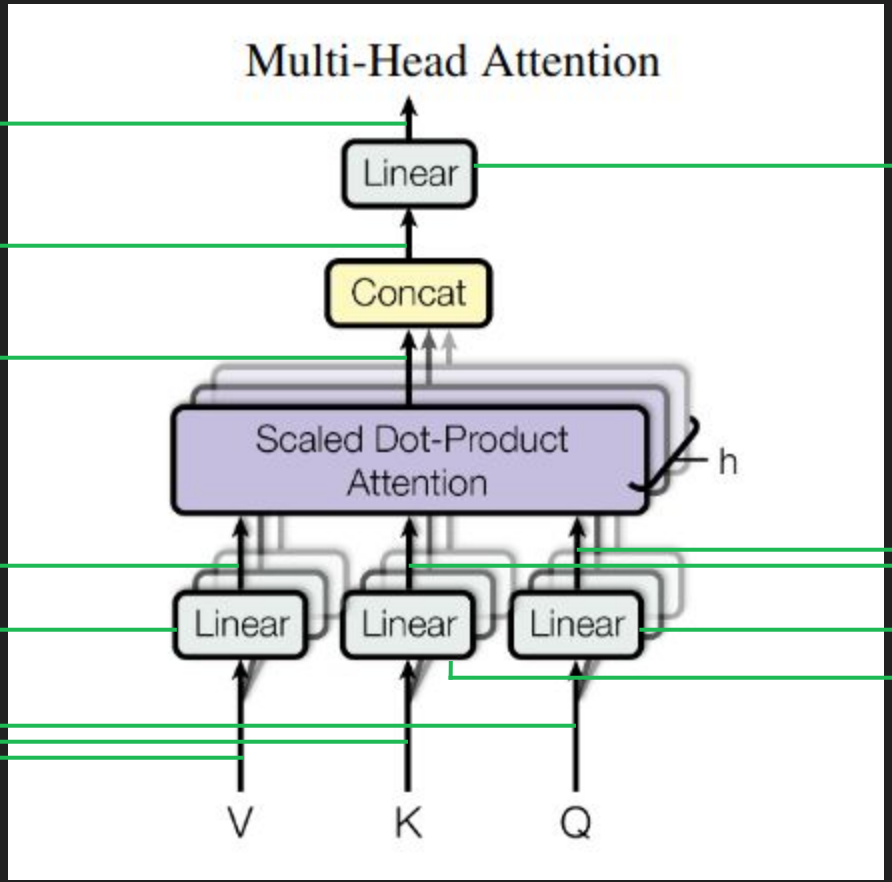

program
-.3
1.1
-.1
.9
de-embedding
projection matrix
=
and
battery
check
down
find
it
log
out
please
program
ran
the
whether
1

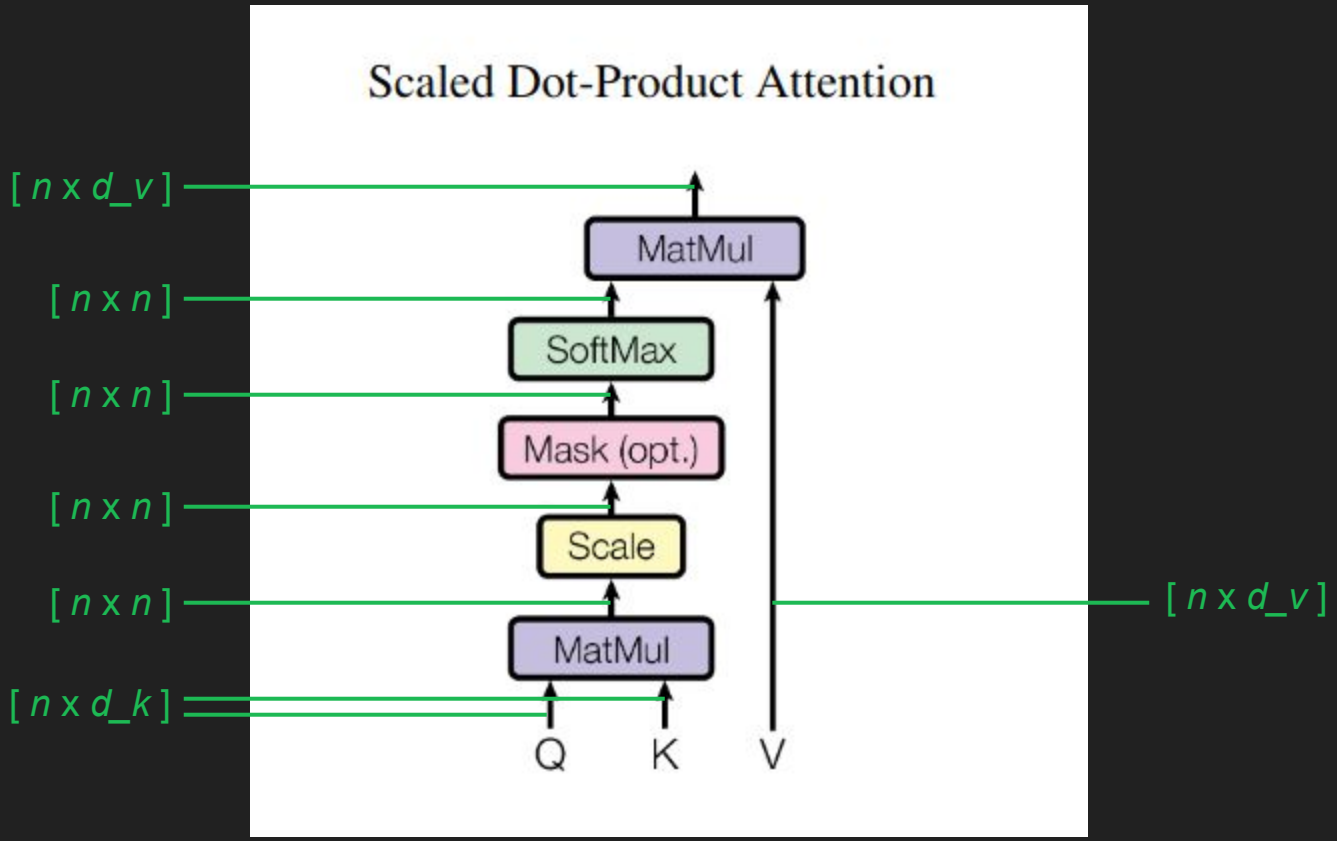

and
battery
check
down
find
it
log
out
please
program
ran
the
whether
-.7
.9
.1
-.2
0
-.7
.8
-.3
-.6
1
.4
-.8
0

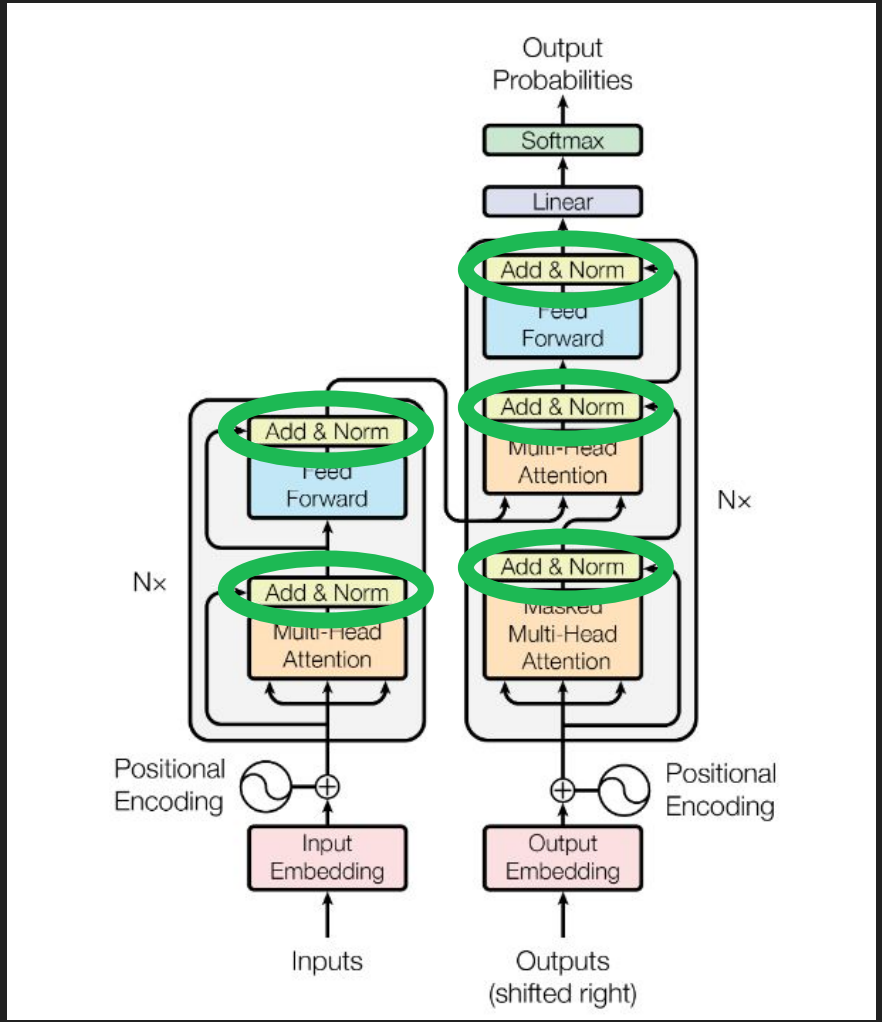

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

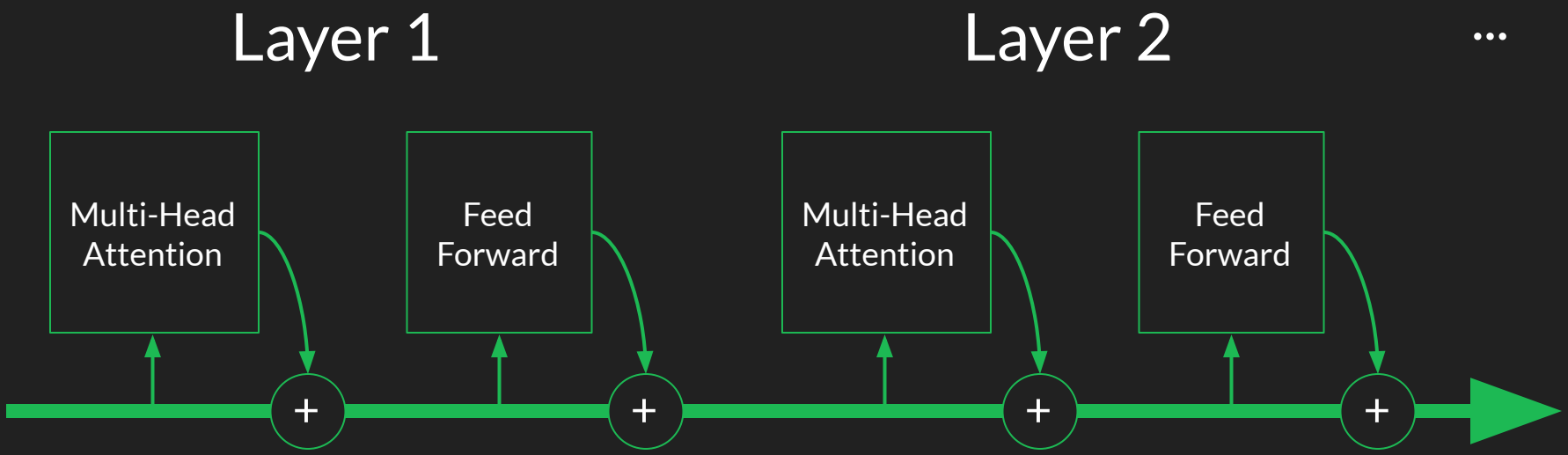

N
d_model
d_model
n
one-hot
word
sequence
N
embedding
transform
=
n
embedded
word
sequence
[n x N]
[N x d_model]
[n x d_model]

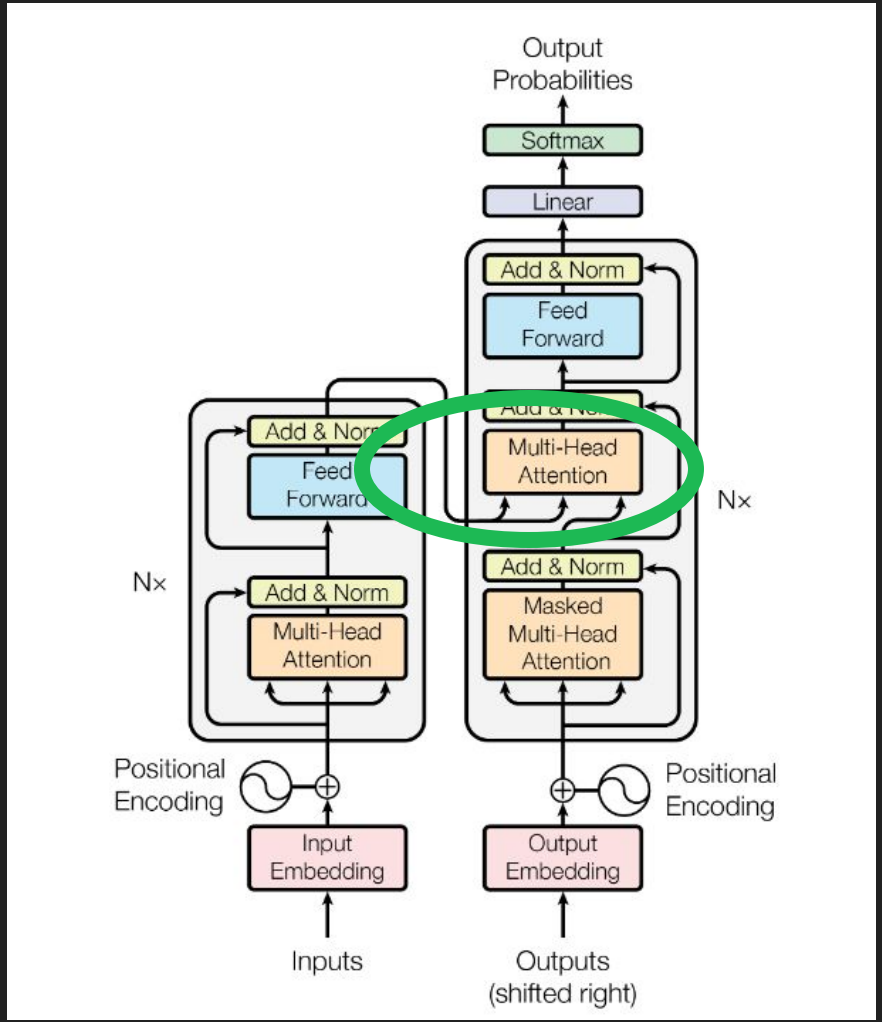

Output Probabilities
[n x N]
Softmax
[n x N]
Linear
[d_model x N]
[n x d_model]
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Add & Norm
Masked Multi-Head Attention
[n x d_model]
Positional Encoding
[n x d_model]
[n x d_model]
Output Embedding
[N x d_model]
[n x N]
Outputs (shifted right)


Multi-Head Attention
[n x d_model]
[n x h*d_v]
[n x d_v]
[n x d_v]
[d_model x d_v]
[n x d_model]
Linear
Concat
Scaled Dot-Product Attention
h
Linear
Linear
Linear
V
K
Q
[h*d_v x d_model]
[n x d_k]
[d_model x d_k]


Scaled Dot-Product Attention
[n x d_v]
[n x n]
[n x n]
[n x n]
[n x n]
[n x d_k]
MatMul
SoftMax
Mask (opt.)
Scale
MatMul
Q
K
V
[n x d_v]


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)


Layer 1
Layer 2
...
Multi-Head Attention
Feed Forward
Multi-Head Attention
Feed Forward
+
+
+
+


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)