

Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM)

RNNs excel at analyzing sequential patterns

Text

Speech

Audio

Video

Physical processes

Anything embedded in time (almost everything)

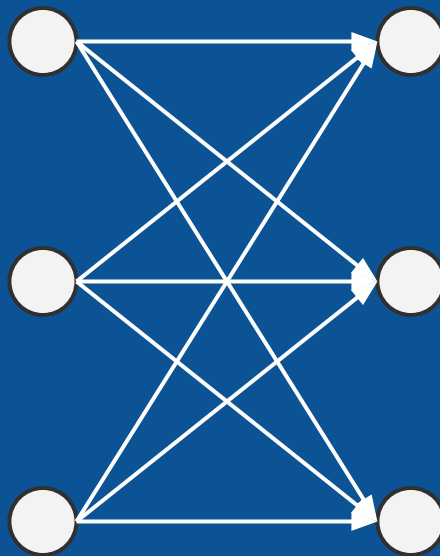
What's for dinner?

Thinking of it in a non-sequential way.

day
of the
week

month
of the
year

late
meeting



pizza



sushi



waffles



What's for dinner?

*But there probably
is a sequential
aspect to it.*

pizza
yesterday

sushi
yesterday

waffles
yesterday



pizza

sushi

waffles



predicted pizza for yesterday

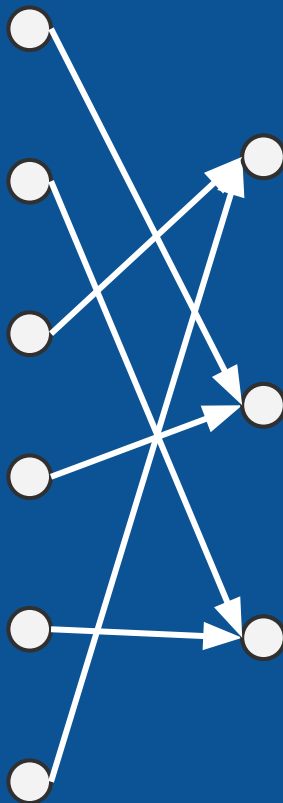
predicted sushi for yesterday

predicted waffles for yesterday

pizza yesterday

sushi yesterday

waffles yesterday



pizza

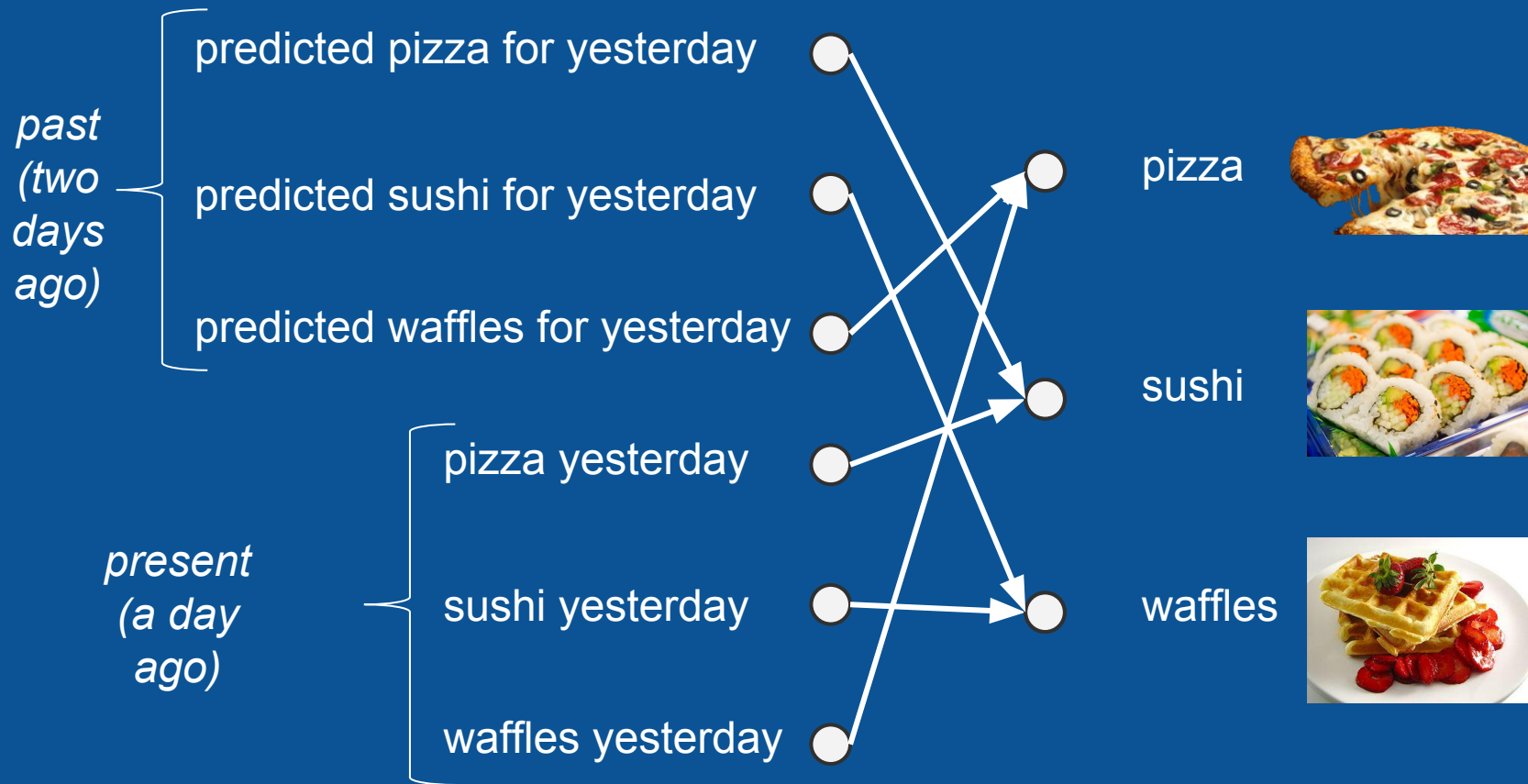


sushi



waffles





predicted pizza for yesterday

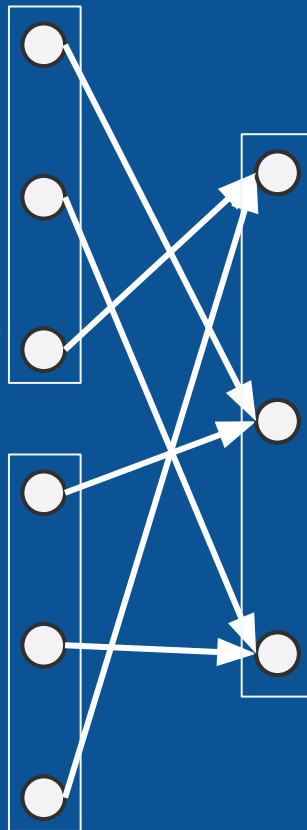
predicted sushi for yesterday

predicted waffles for yesterday

pizza yesterday

sushi yesterday

waffles yesterday



pizza



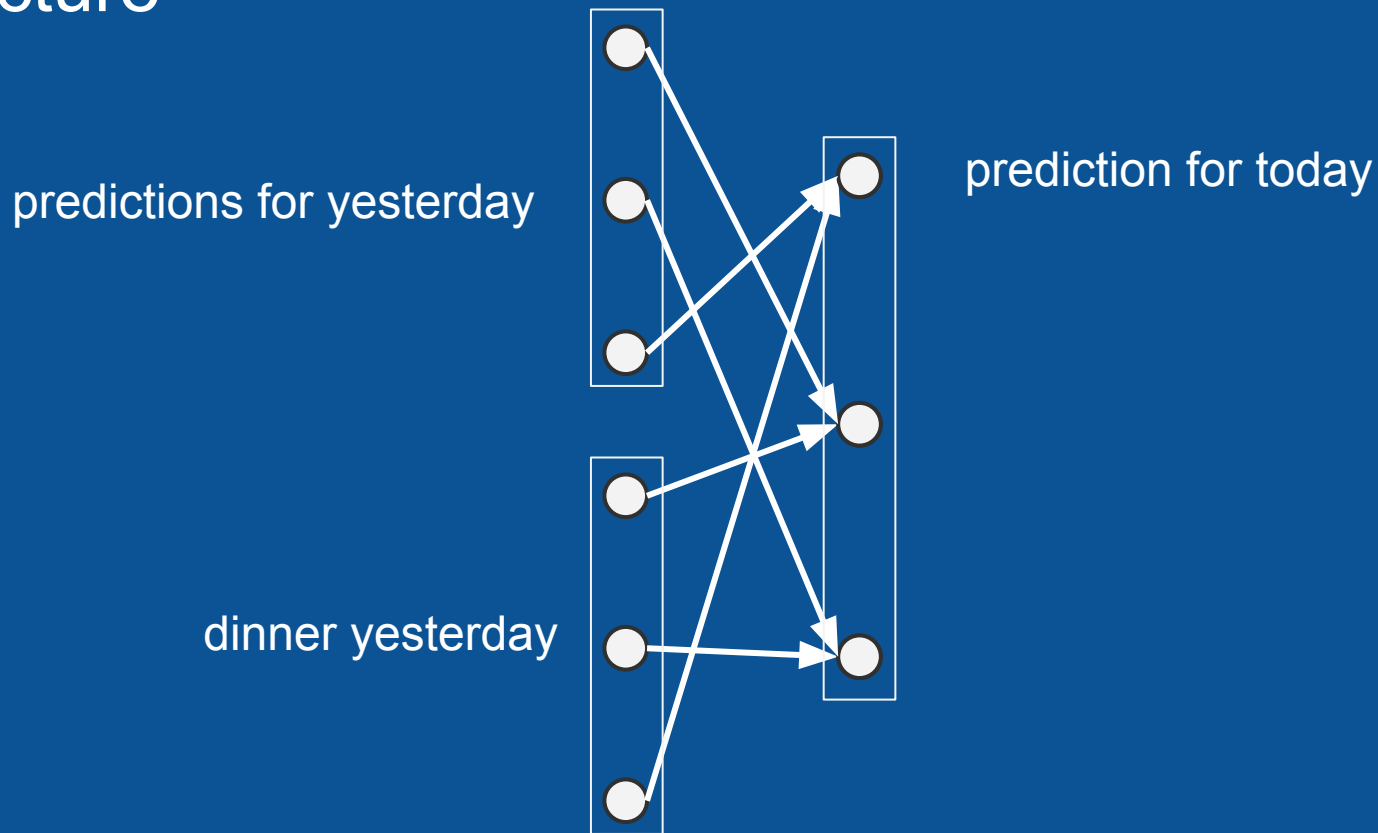
sushi



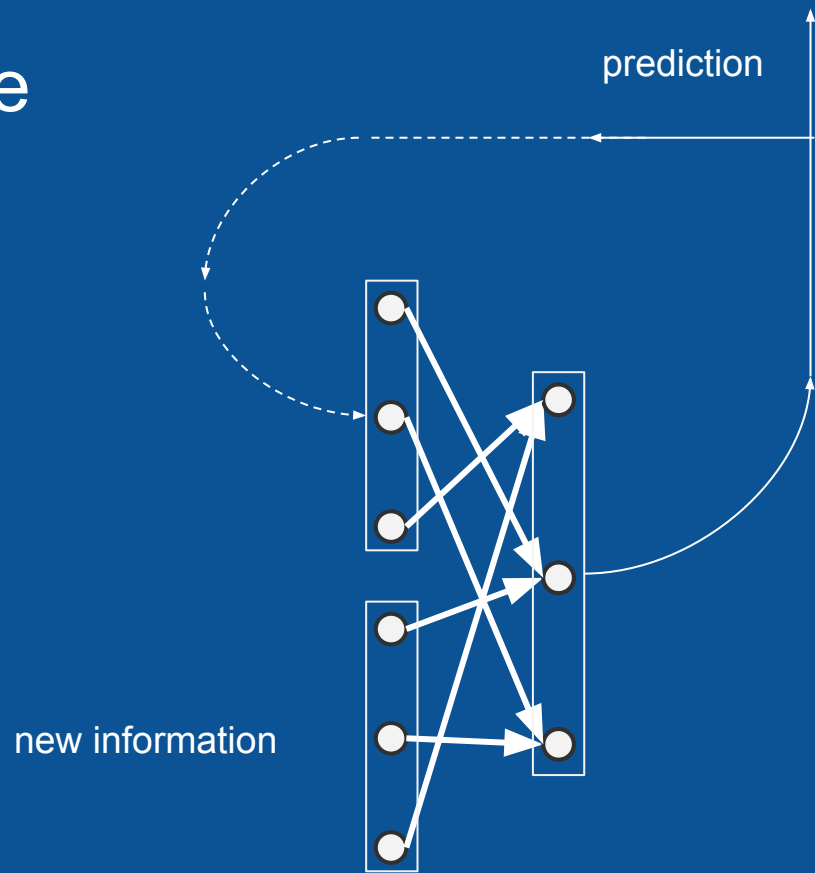
waffles



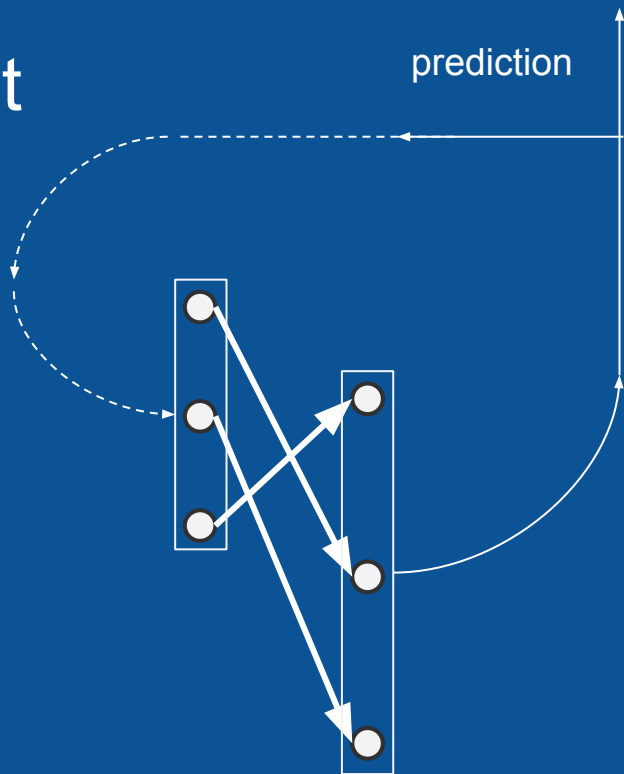
Architecture



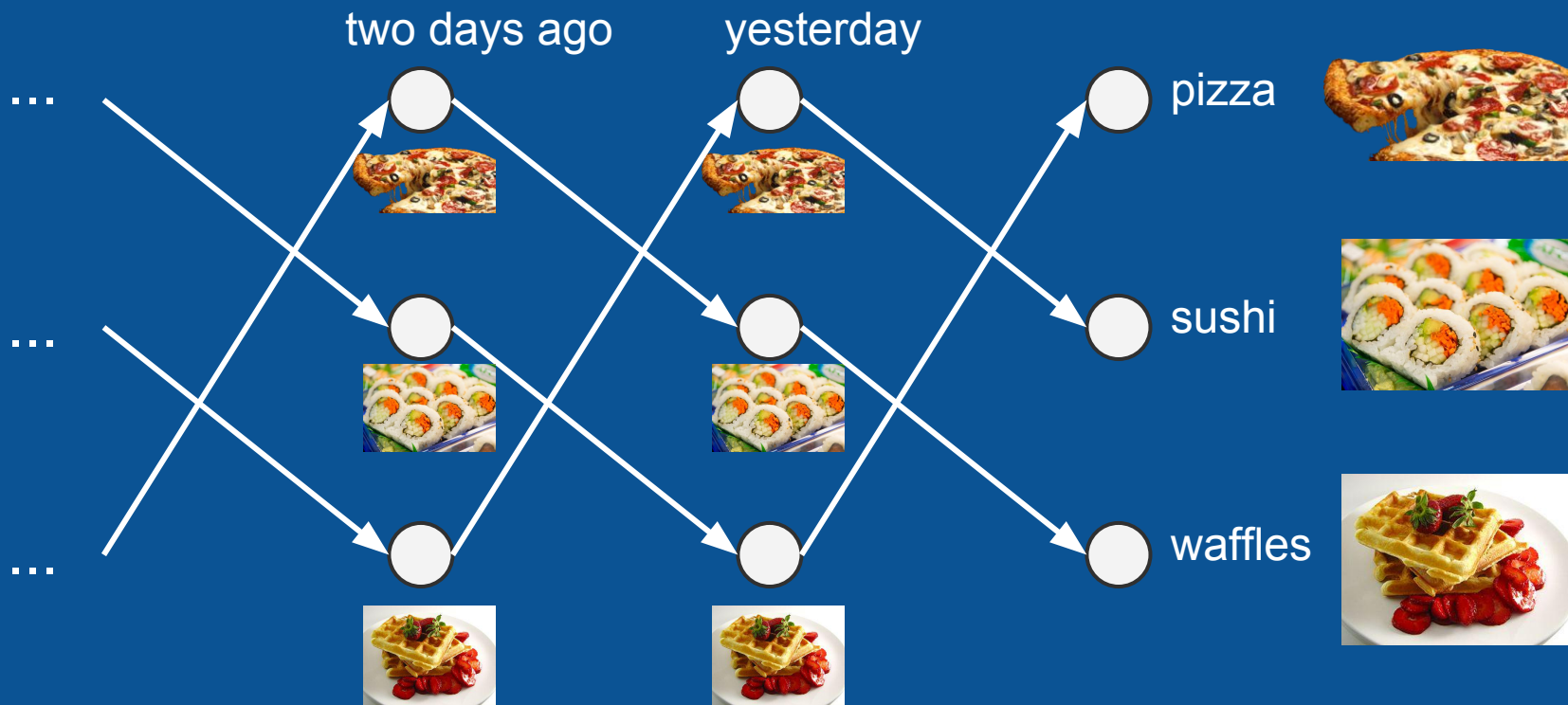
Architecture



Unrolling the past



Unrolled predictions



Another example: Write a children's book

Training data:

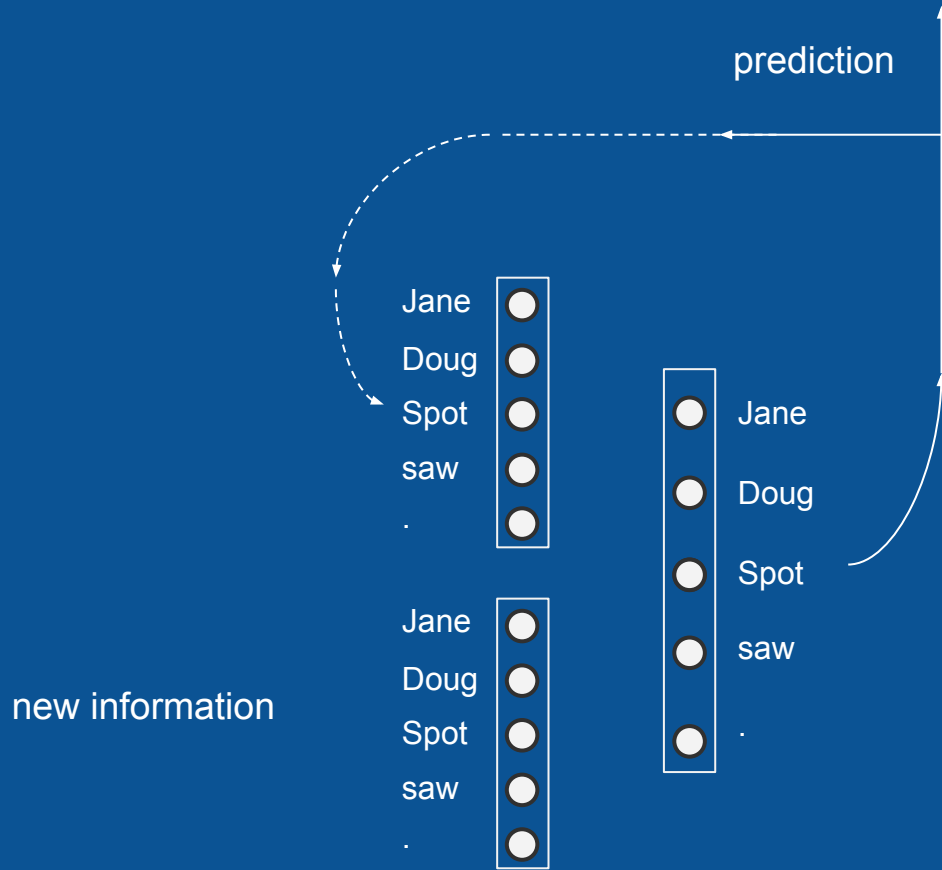
Doug saw Jane.

Jane saw Spot.

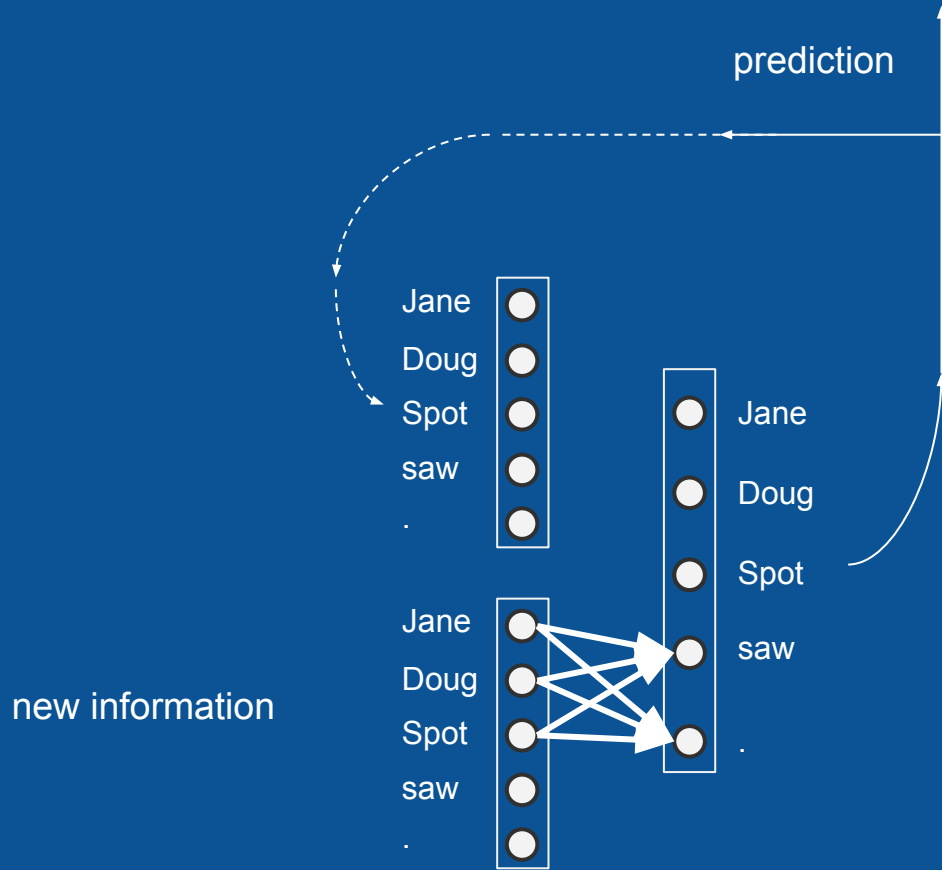
Spot saw Doug.

...

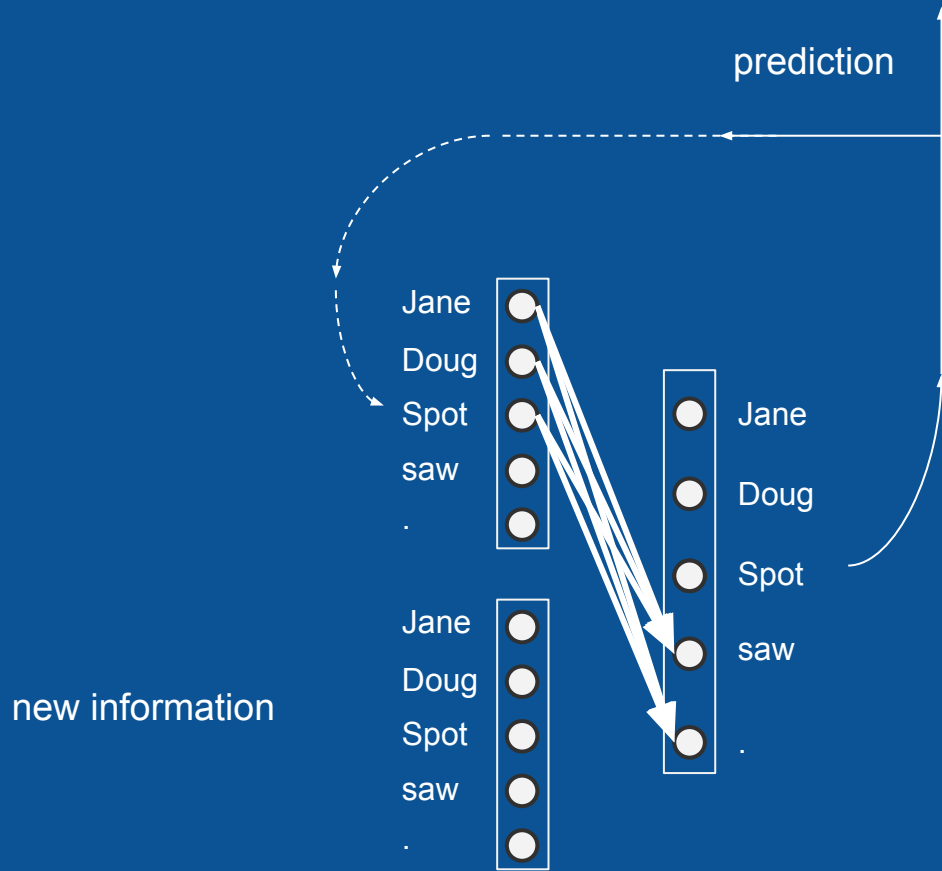
Your dictionary is small: {Doug, Jane, Spot, saw, .}

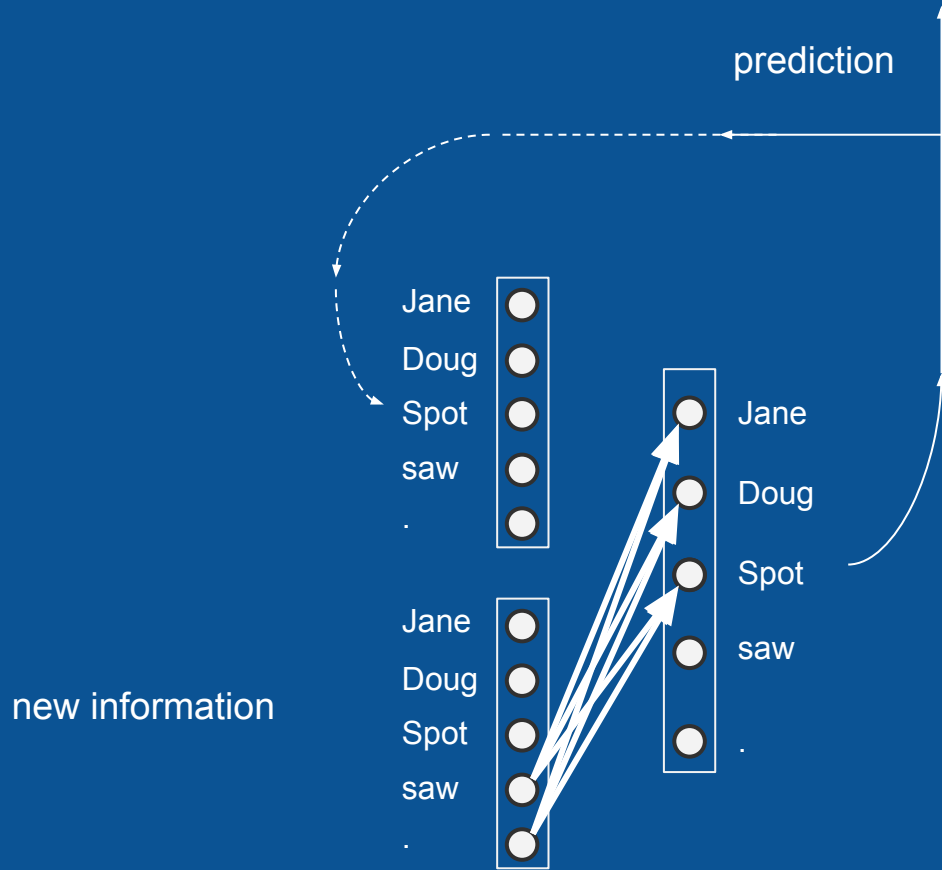


All the words can be new information, and all the words can be the prediction.



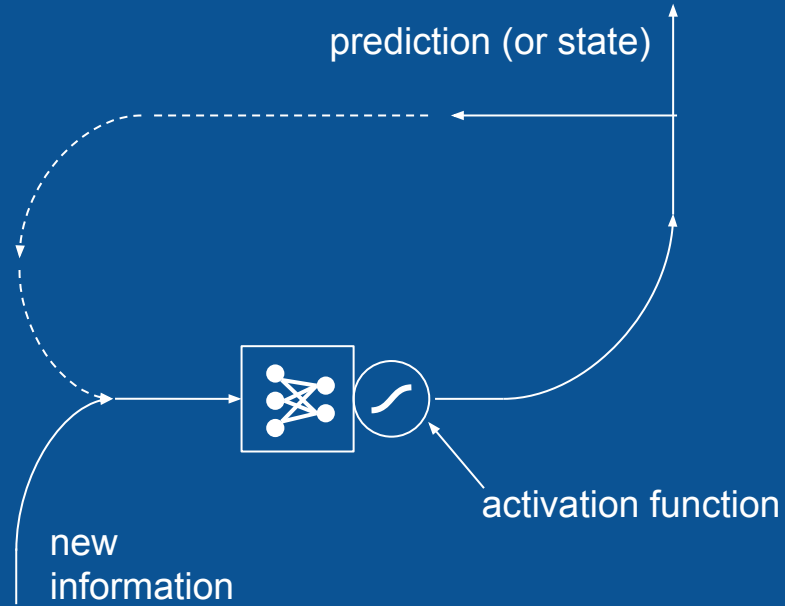
However, based on training corpus, after one of the names the next “word” is highly likely to be “saw” or “.”



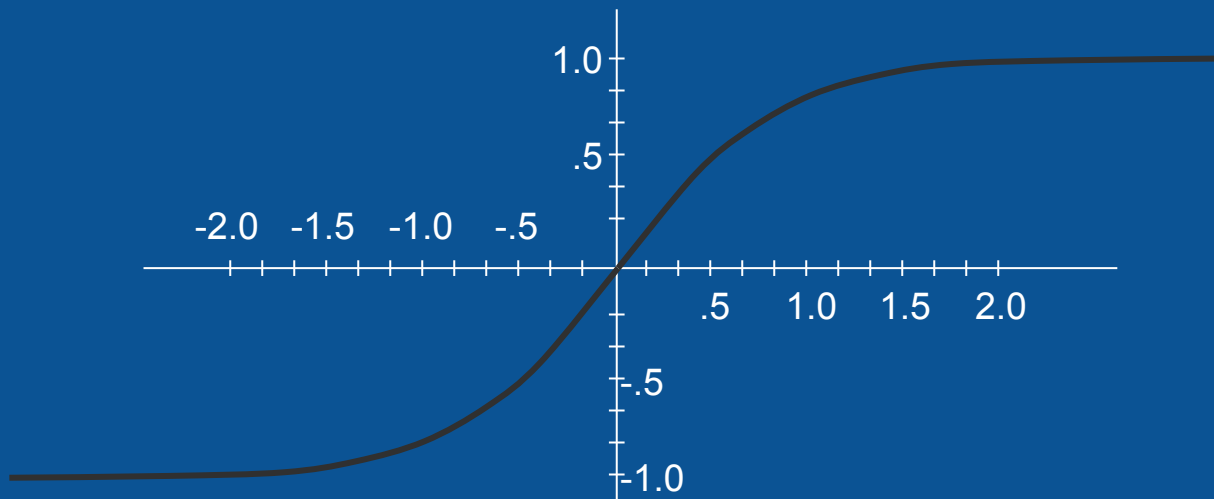


The converse is true, too.

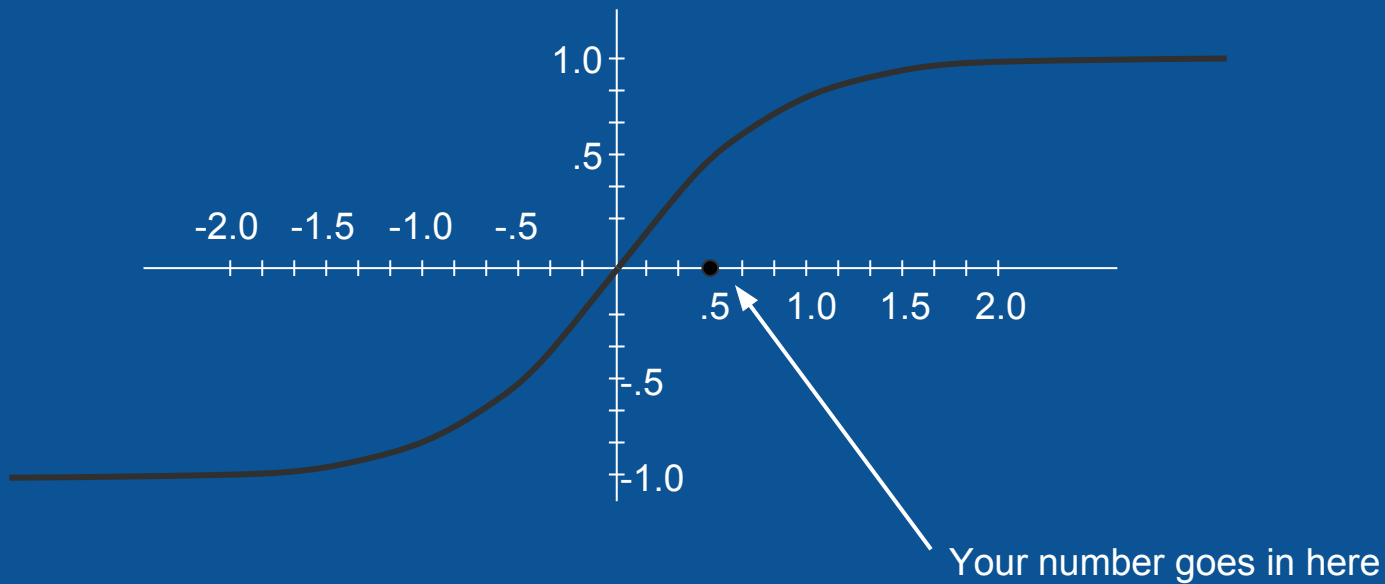
recurrent neural network



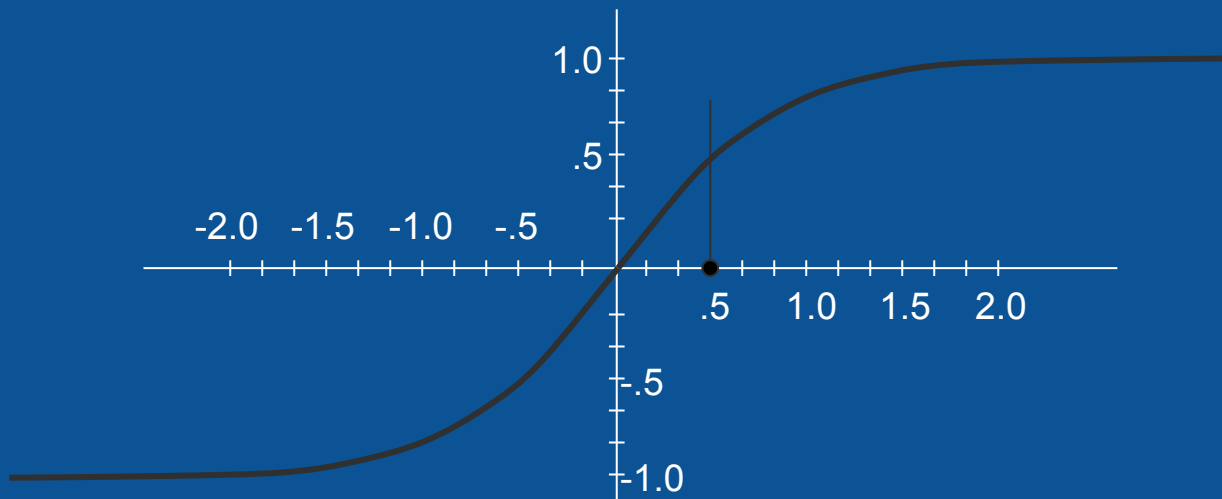
Hyperbolic tangent (tanh) activation function



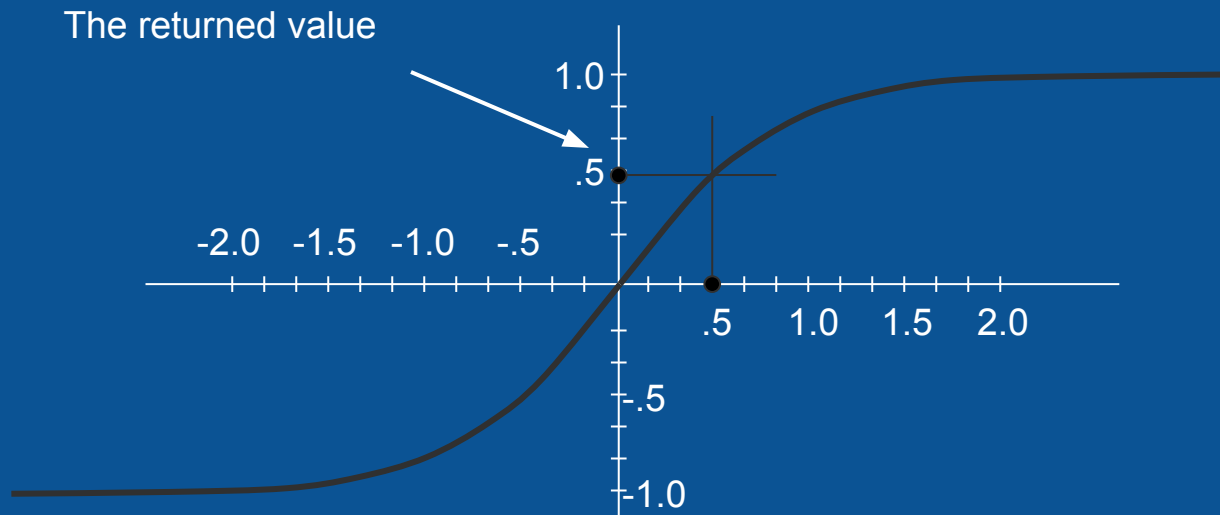
tanh activation function



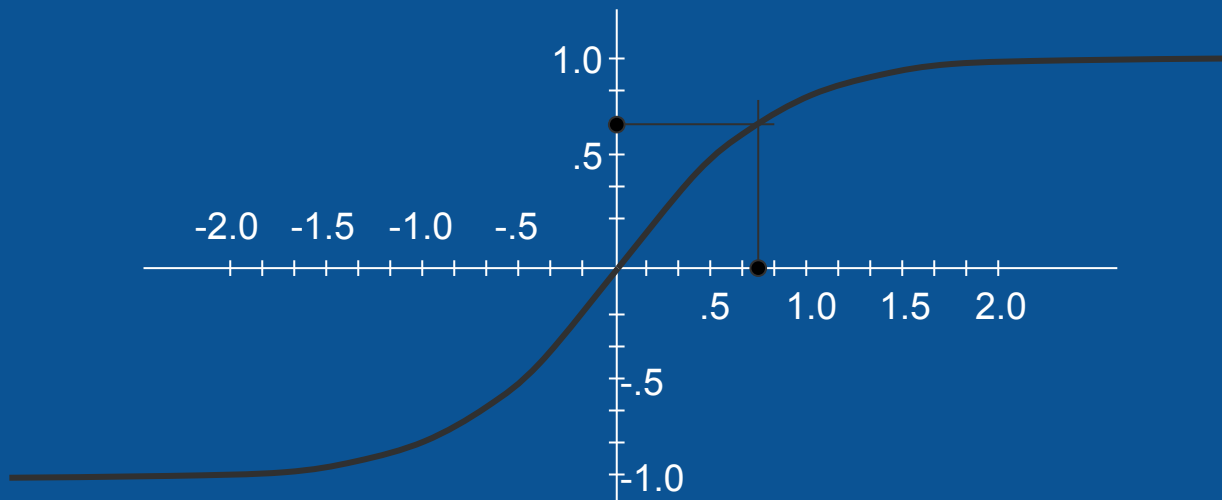
tanh activation function



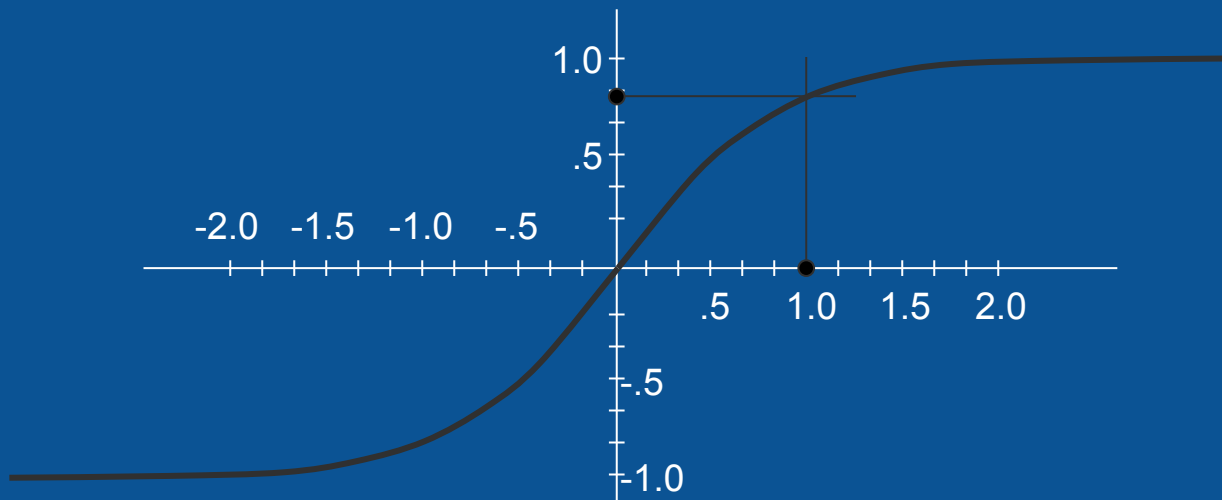
tanh activation function



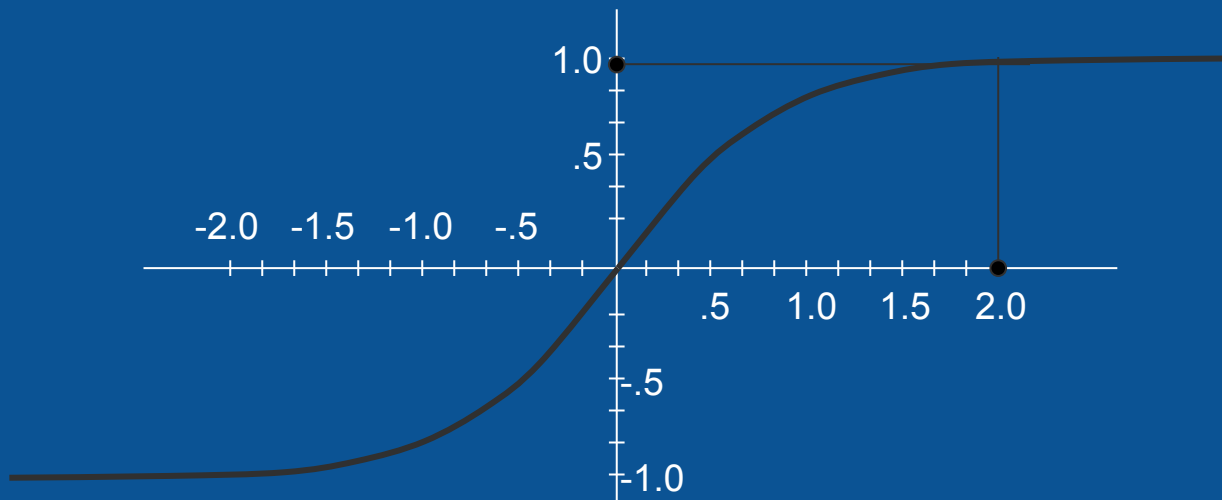
tanh activation function



tanh activation function

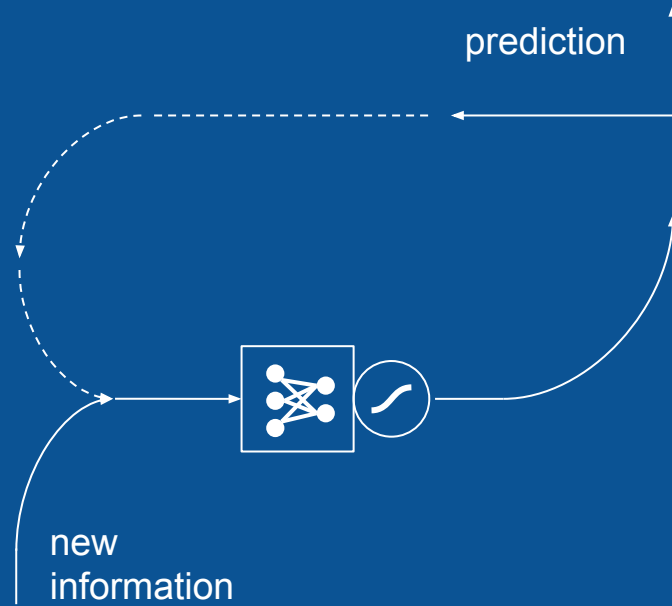


tanh activation function



No matter what you start with, the answer stays between -1 and 1.

recurrent neural network



Mistakes a simple RNN can make

Our predicted story:

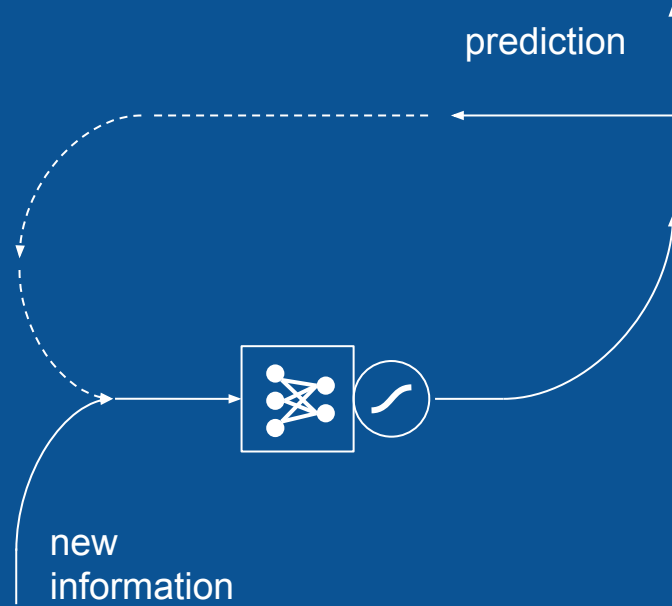
Doug saw Doug.

Jane saw Spot saw Doug saw ...

Spot. Doug. Jane.

*We need to remember farther back in time, maybe make some things off-limits.
Enter the LSTM.*

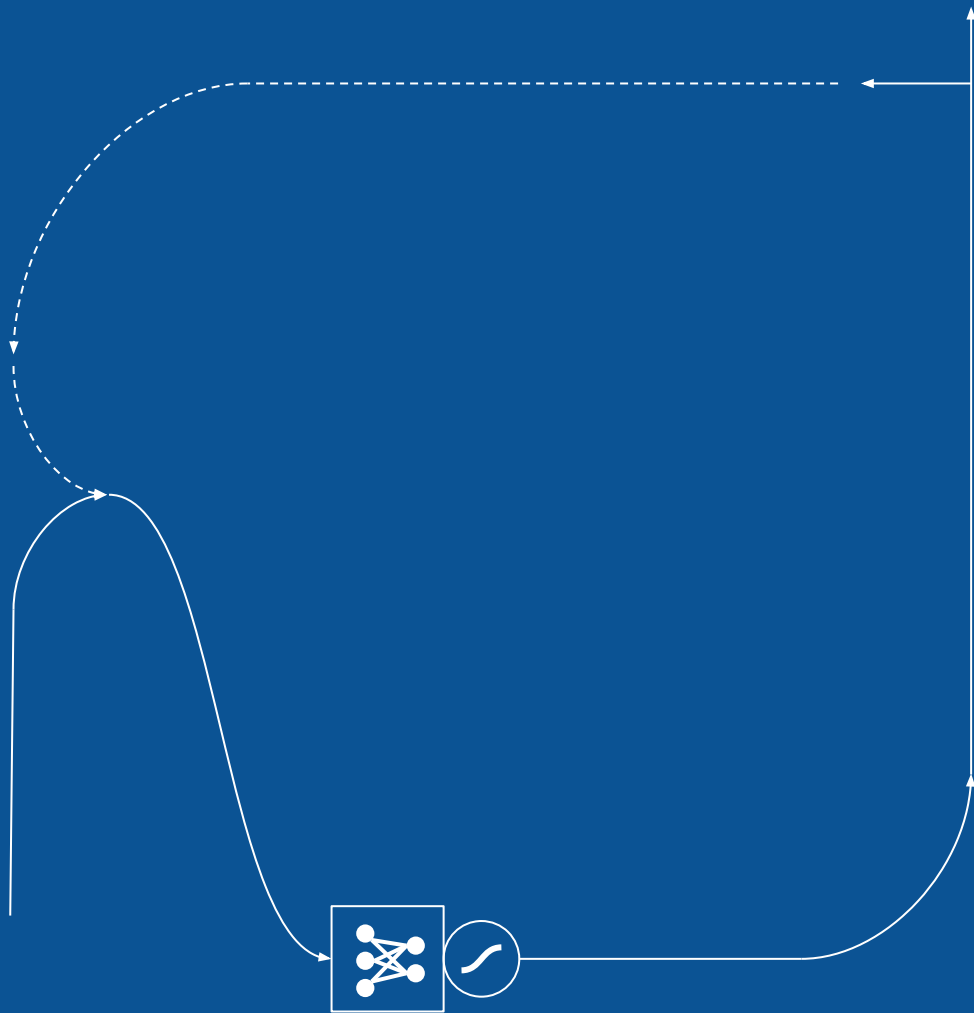
recurrent neural network (vanilla)



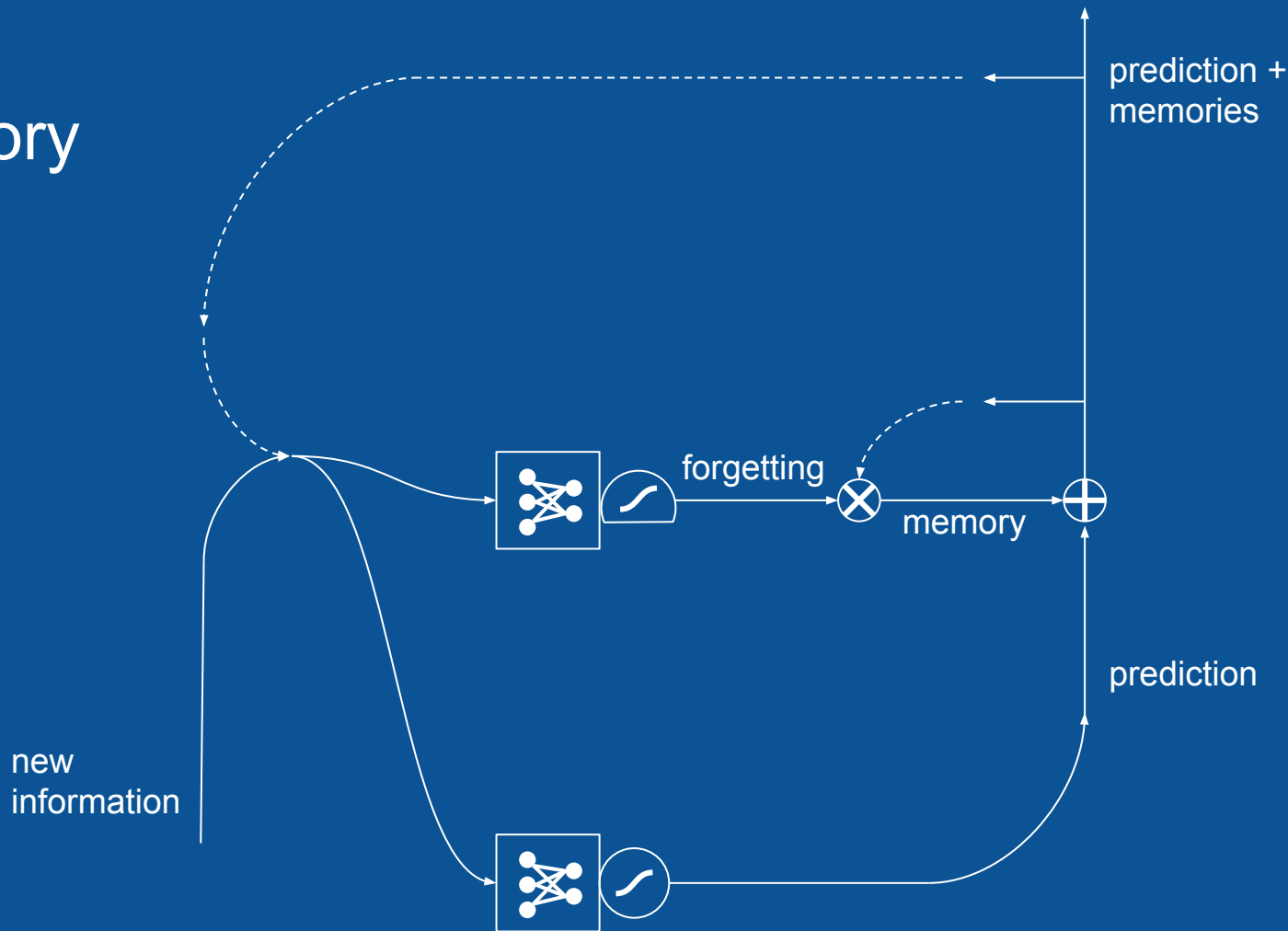
recurrent neural network (vanilla)

new
information

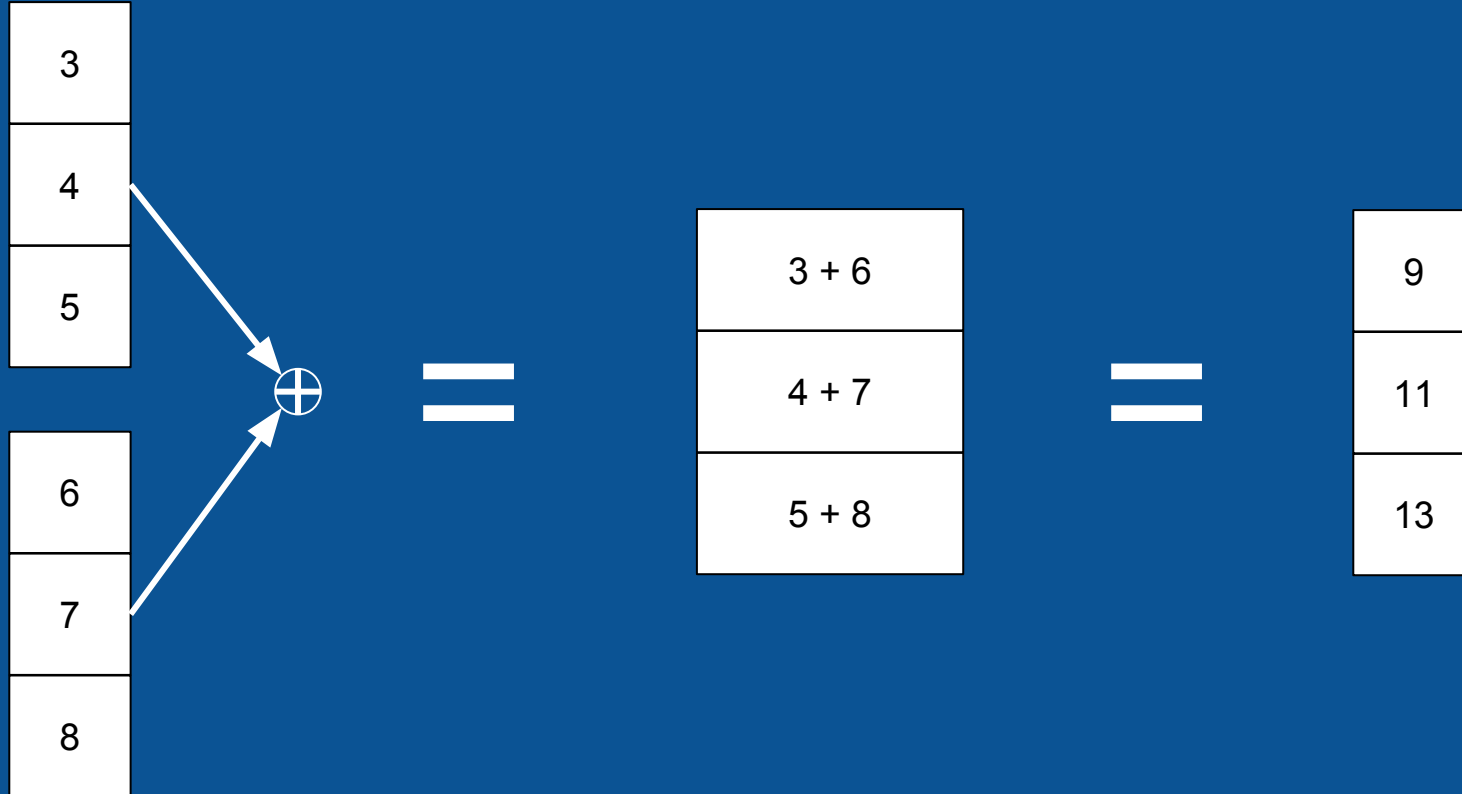
prediction



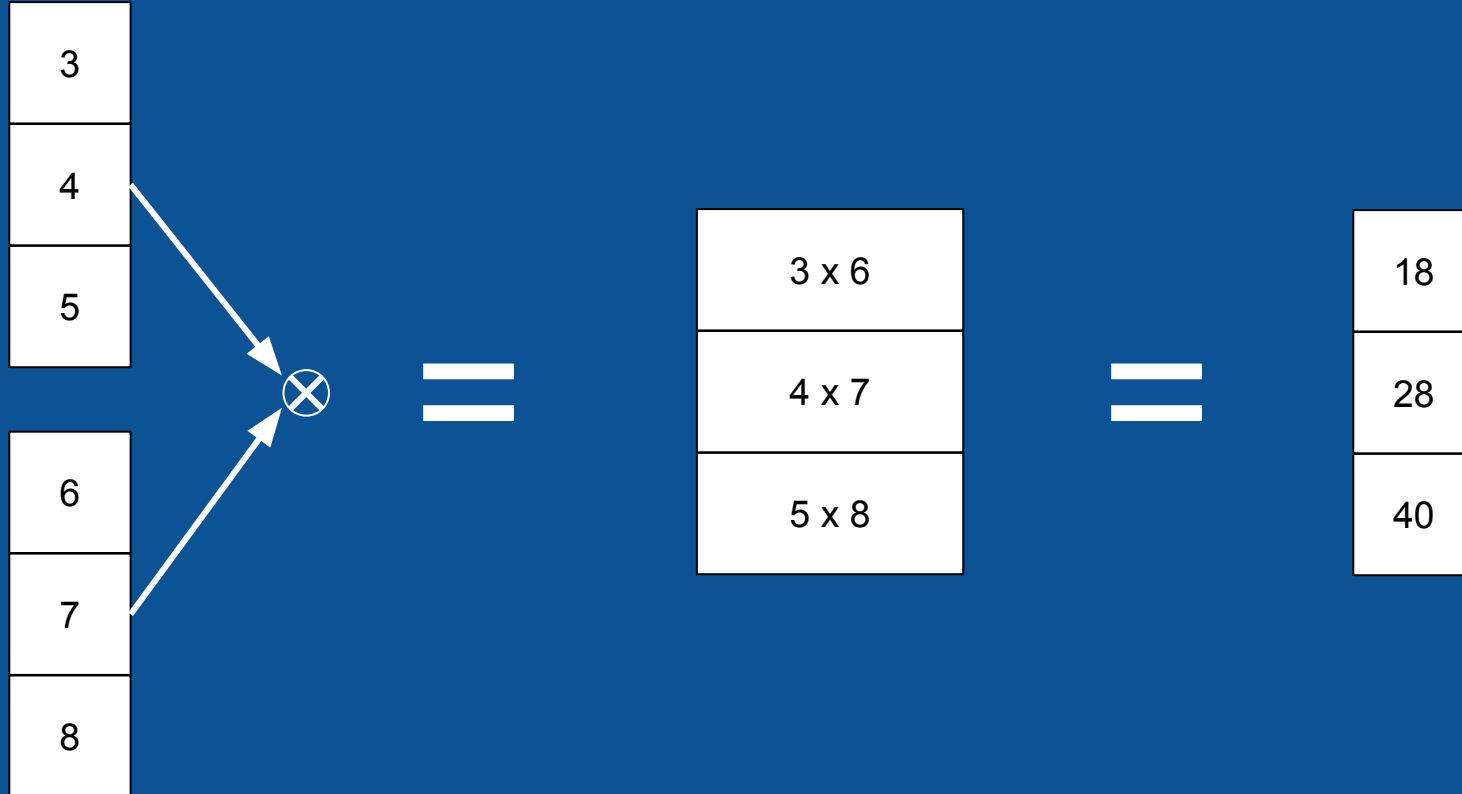
add
memory
and
gates



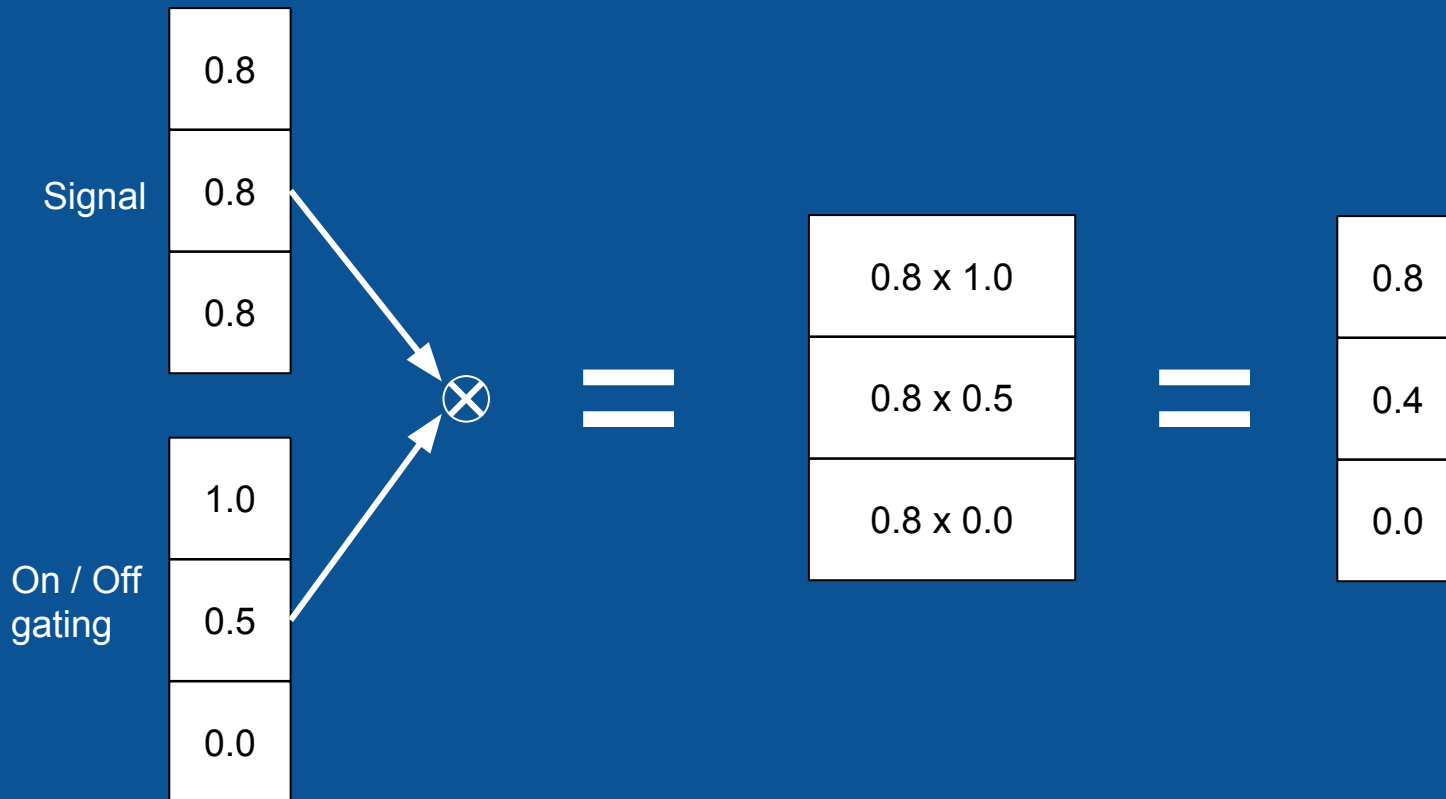
Plus junction: element-by-element addition



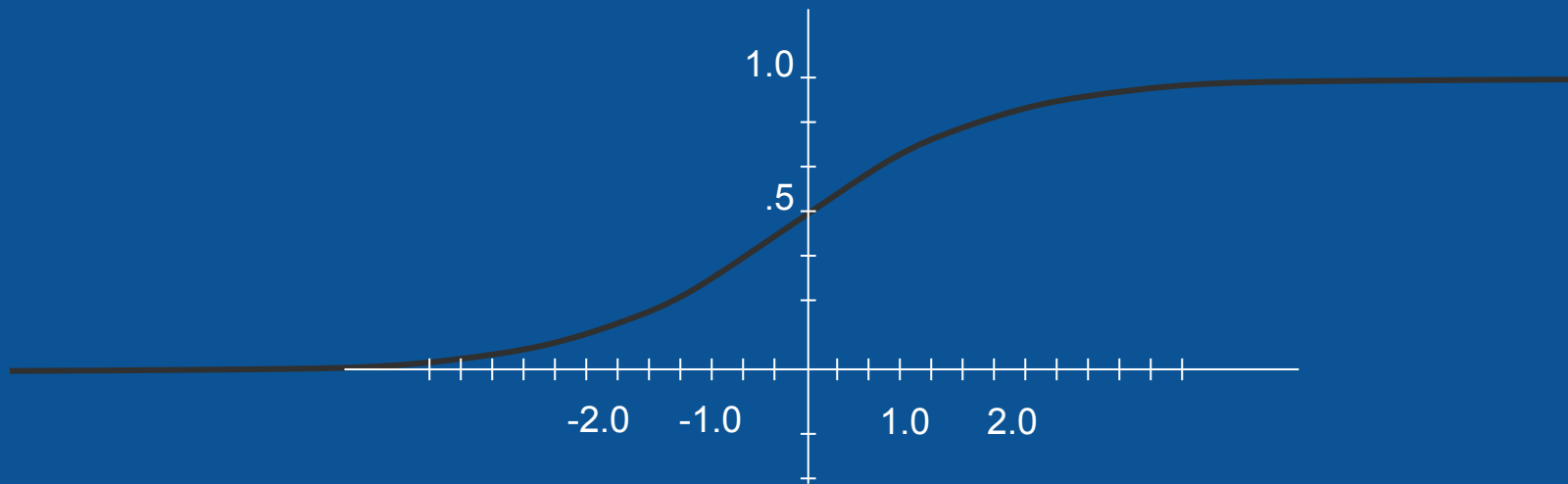
Times junction: element-by-element multiplication



Gating

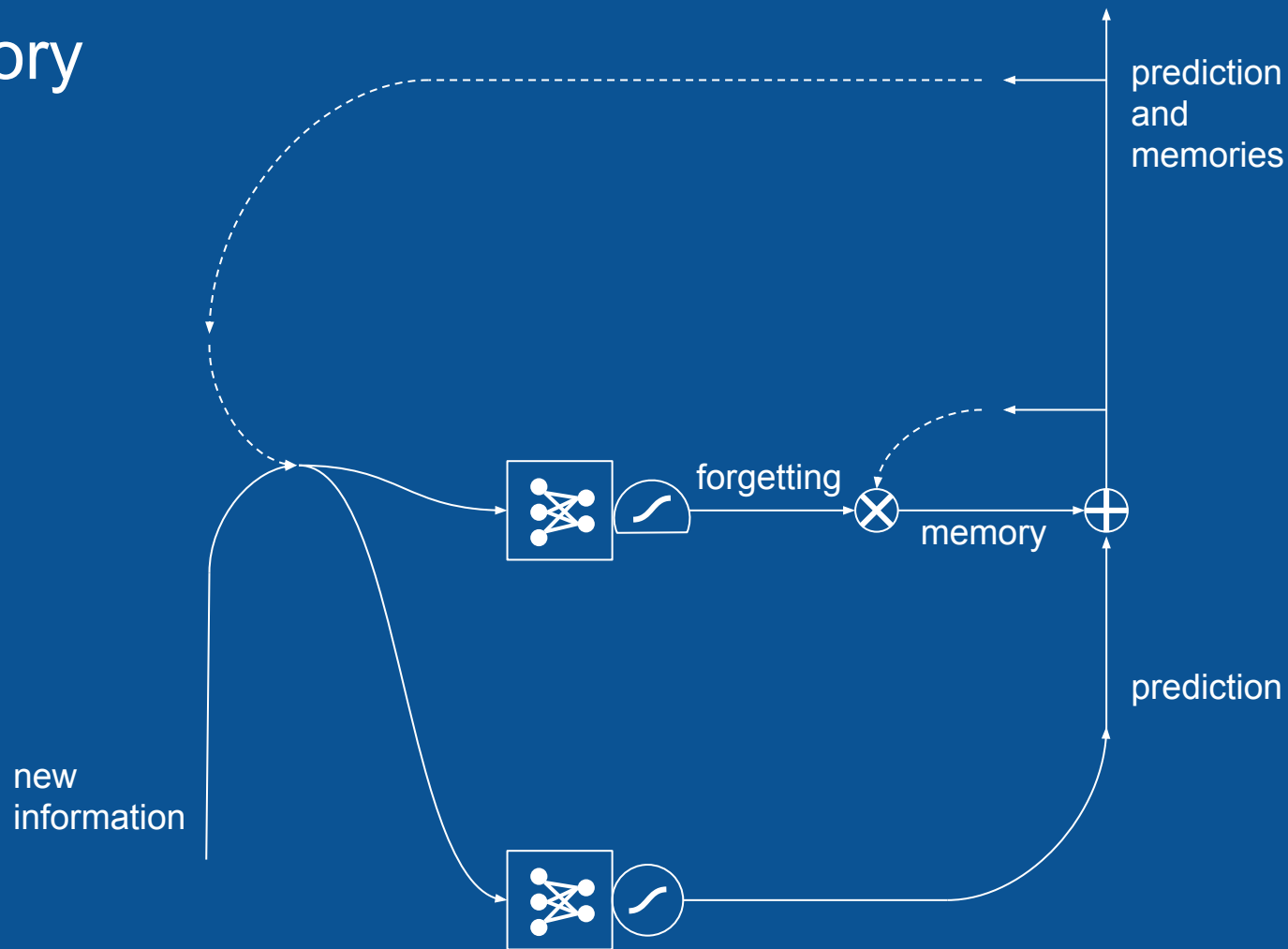


Logistic (sigmoid) activation function

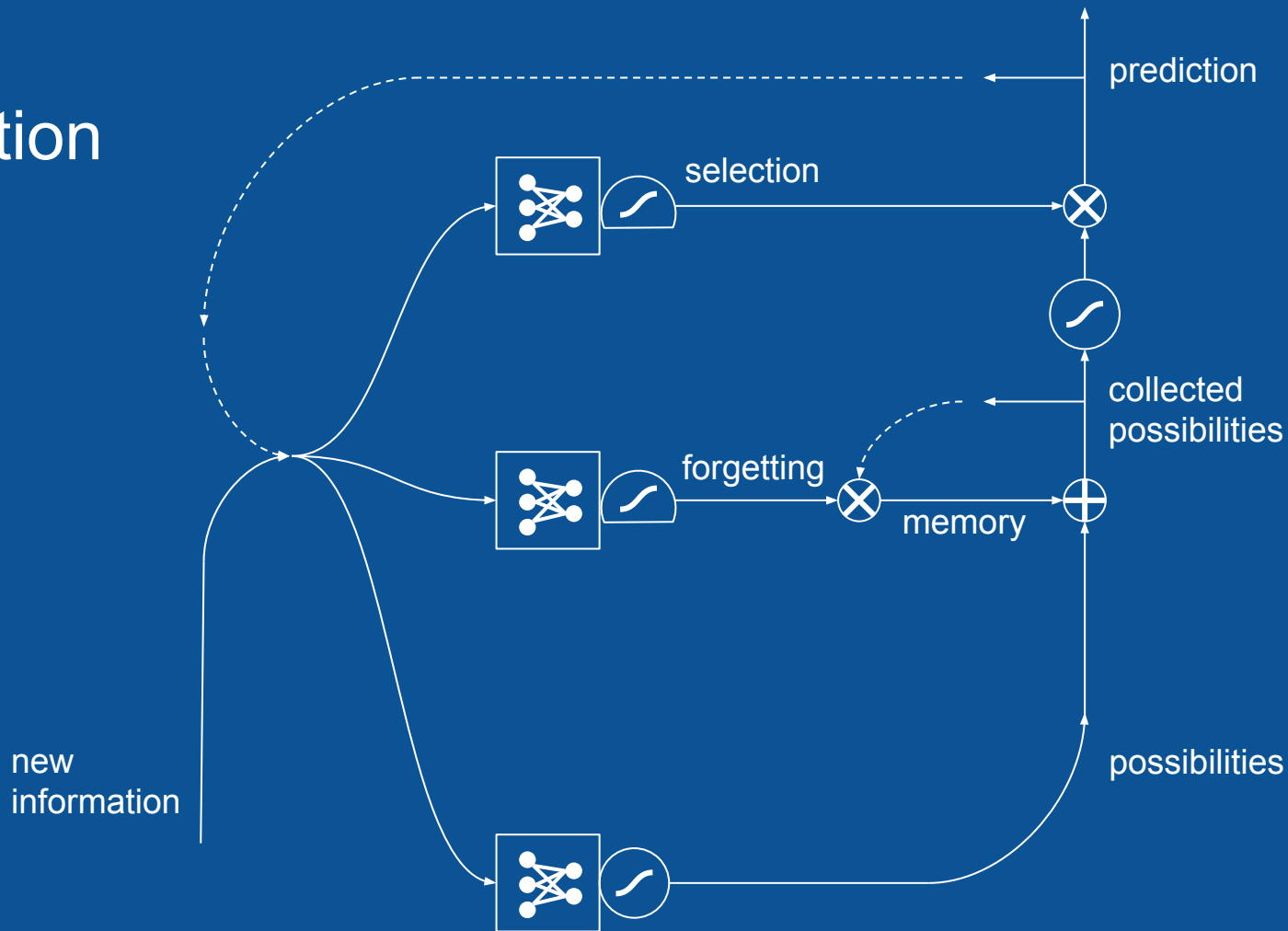


No matter what you start with, the answer stays between 0 and 1.

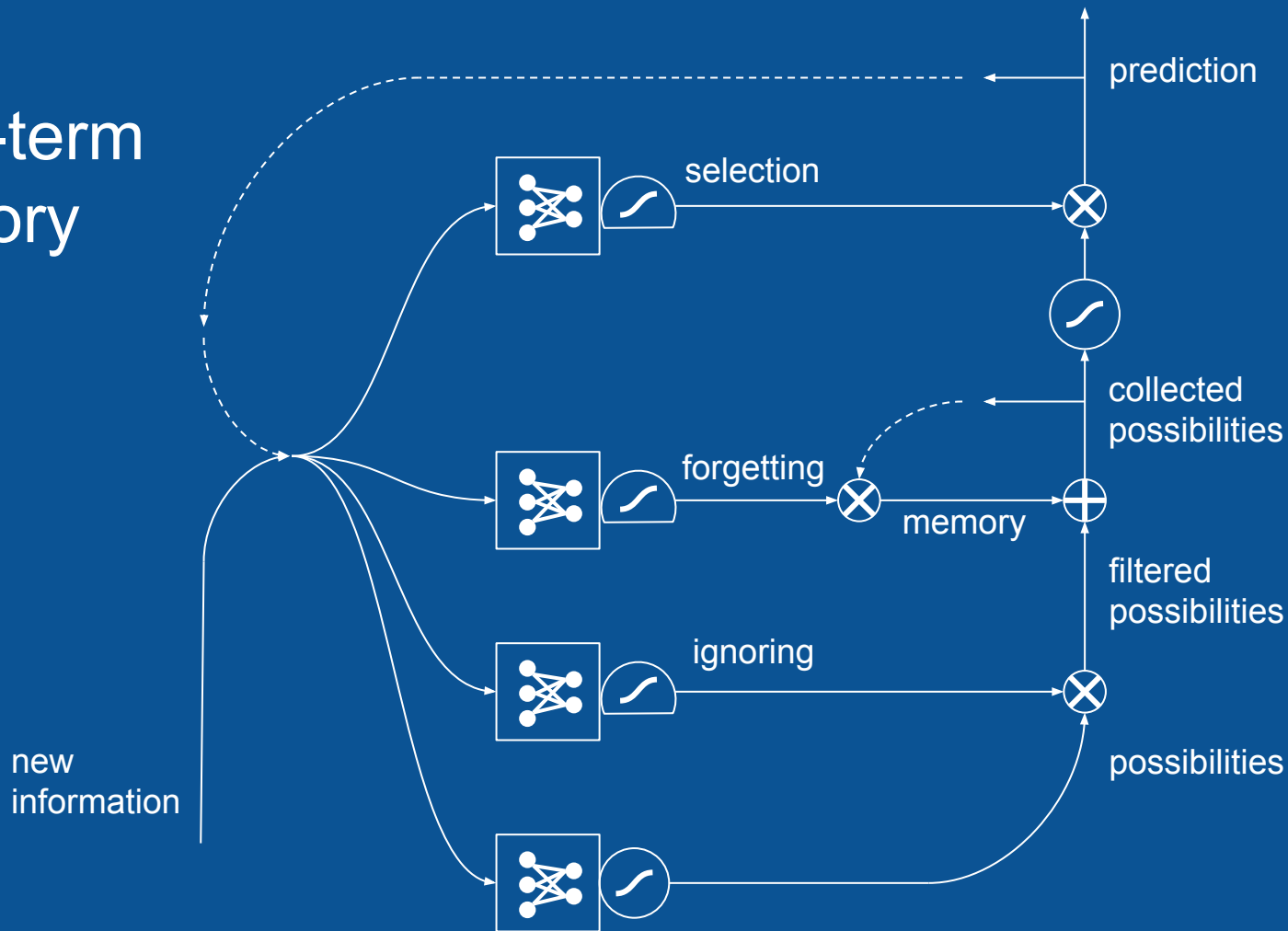
memory



add
selection

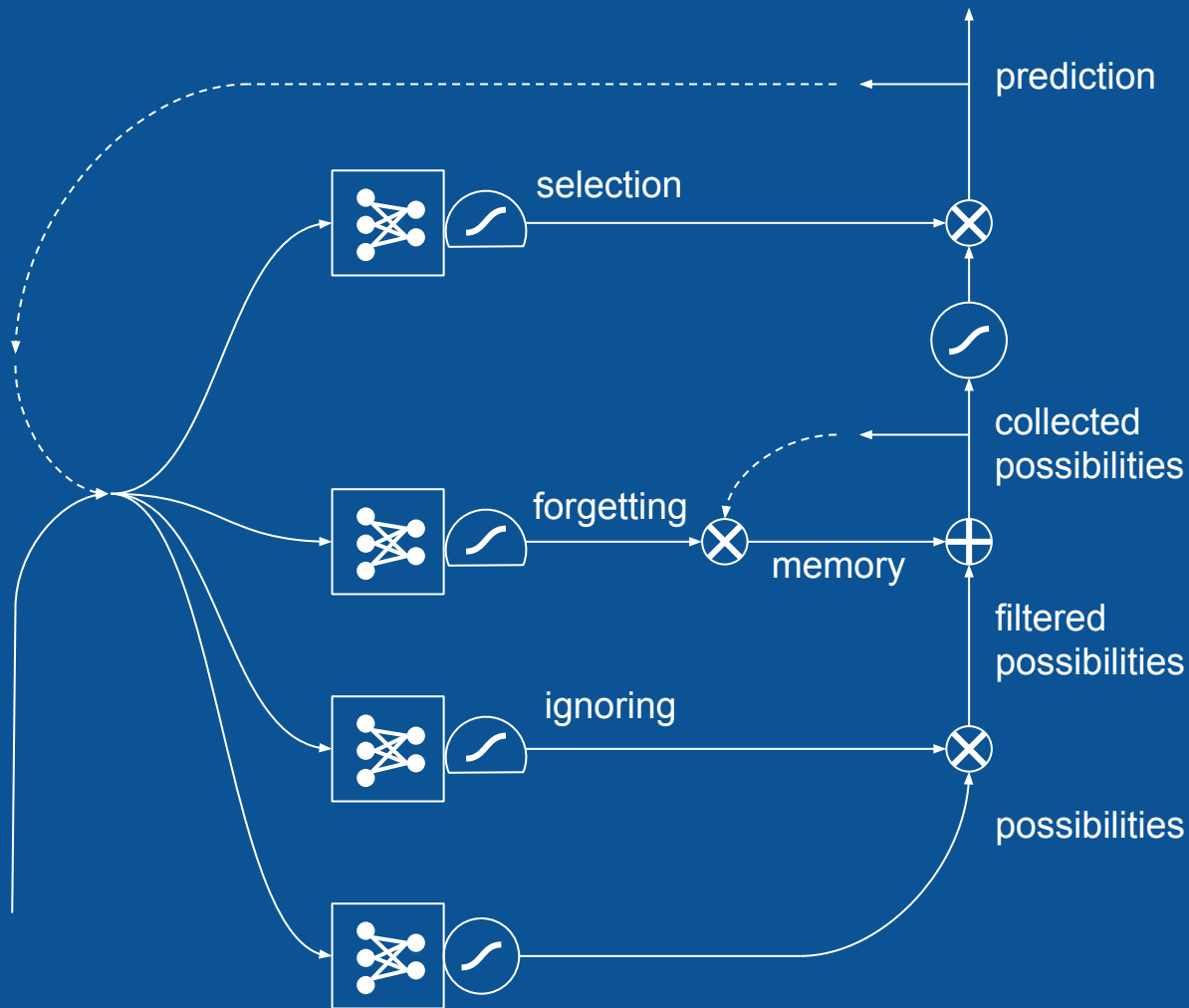


long short-term memory



long short-term memory

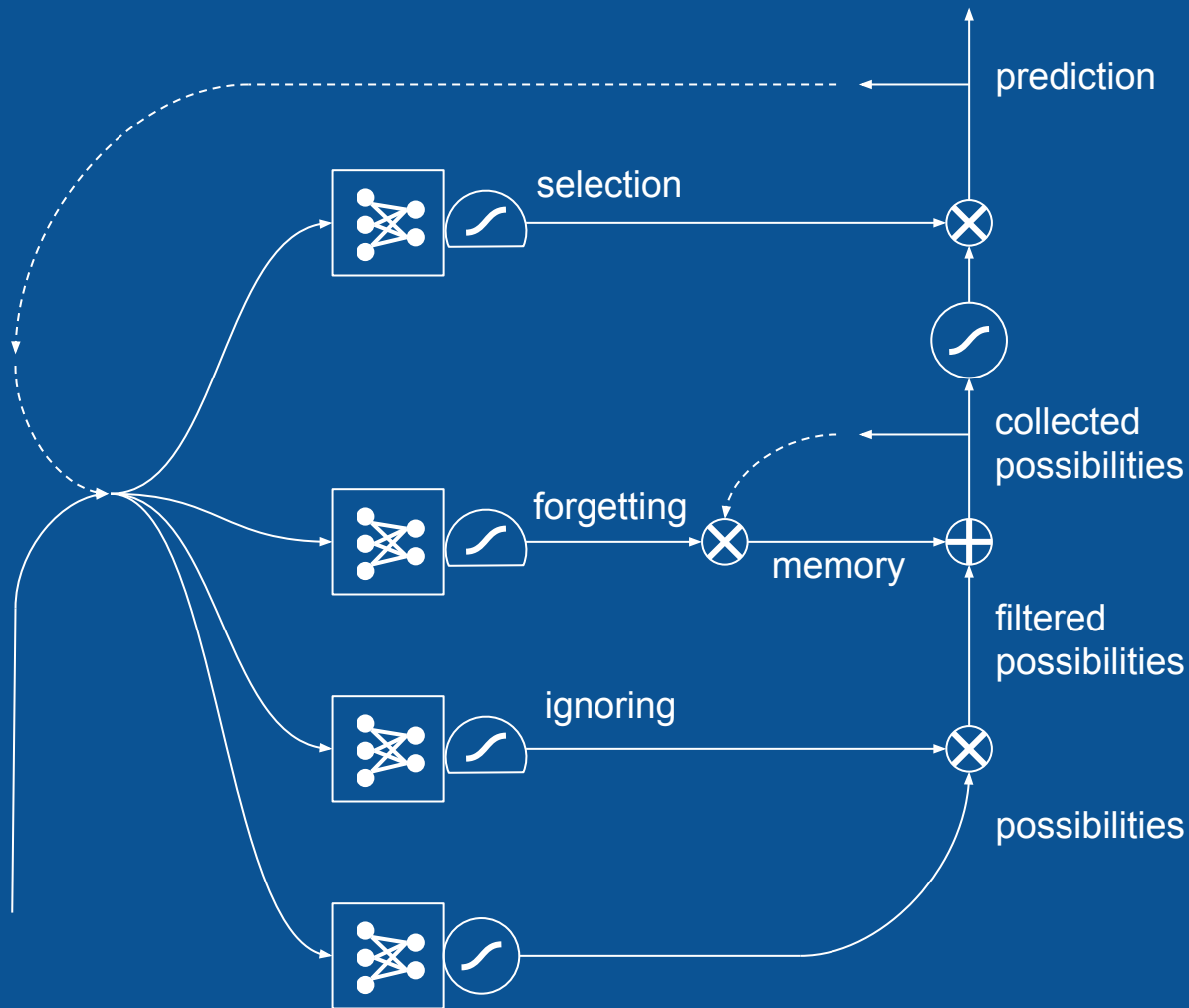
Jane saw Spot.
Doug ...



long
short-term
memory

Doug,
Jane,
Spot

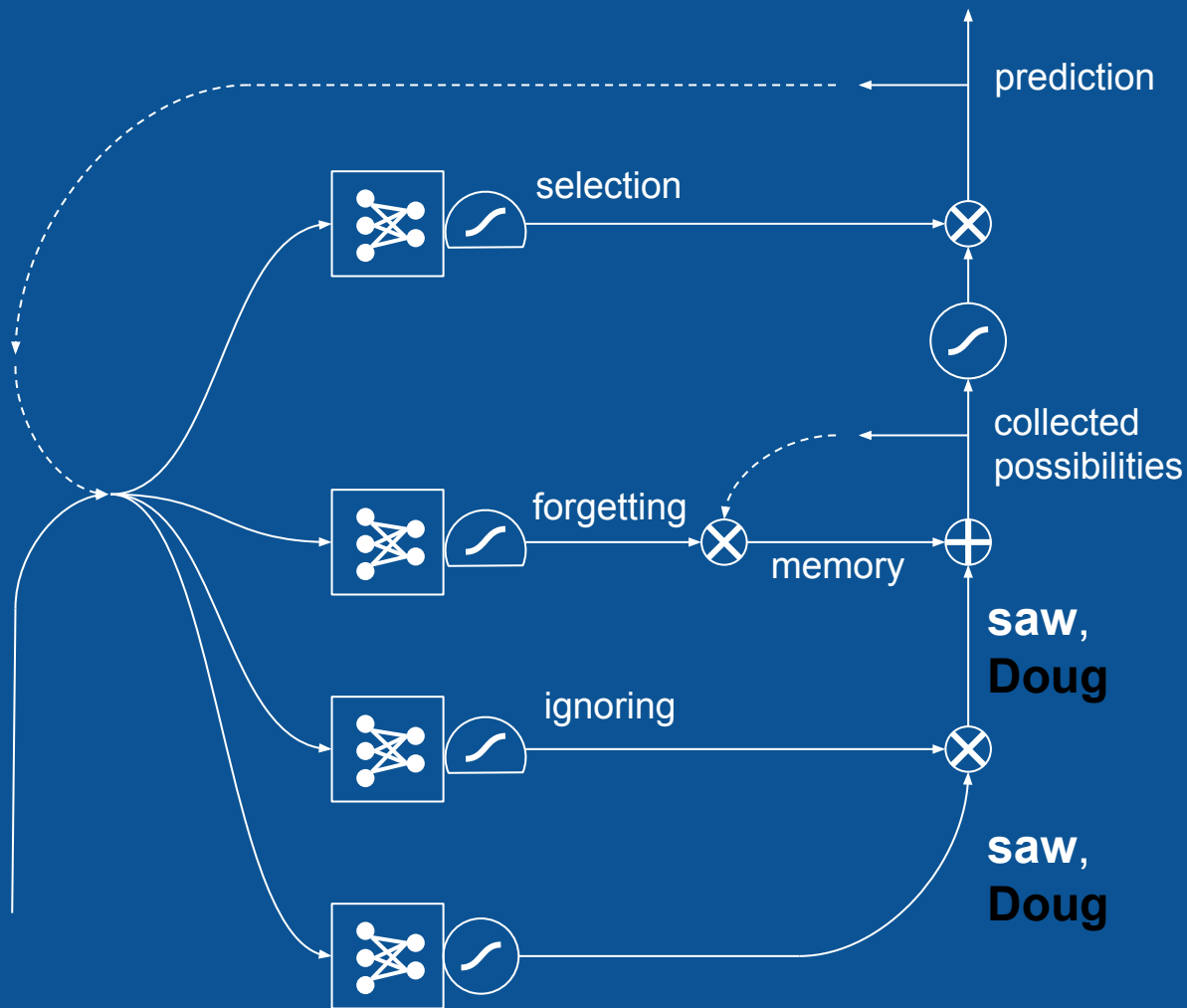
Jane saw Spot.
Doug ...



long
short-term
memory

Doug,
Jane,
Spot

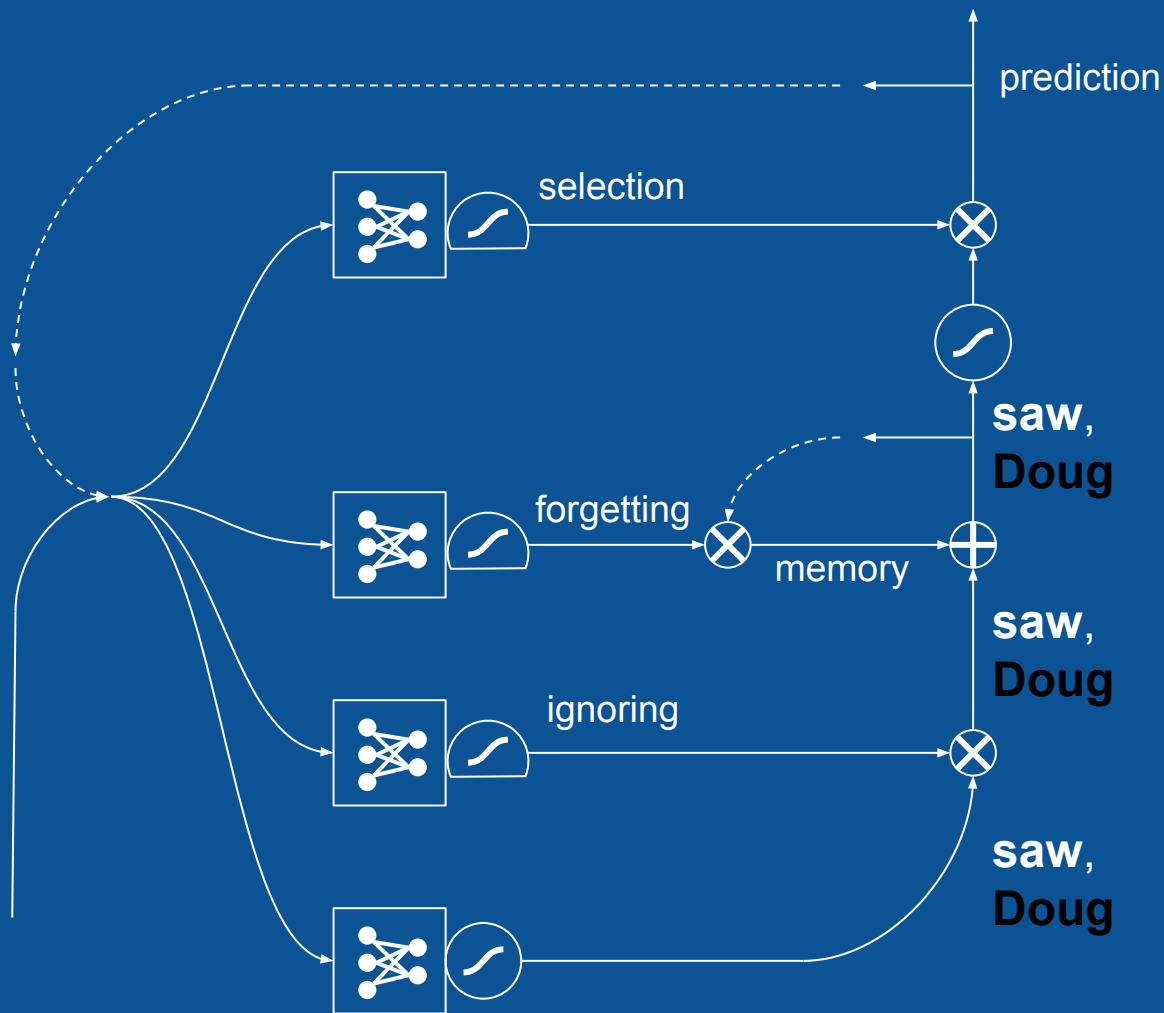
Jane saw Spot.
Doug ...



long
short-term
memory

Doug,
Jane,
Spot

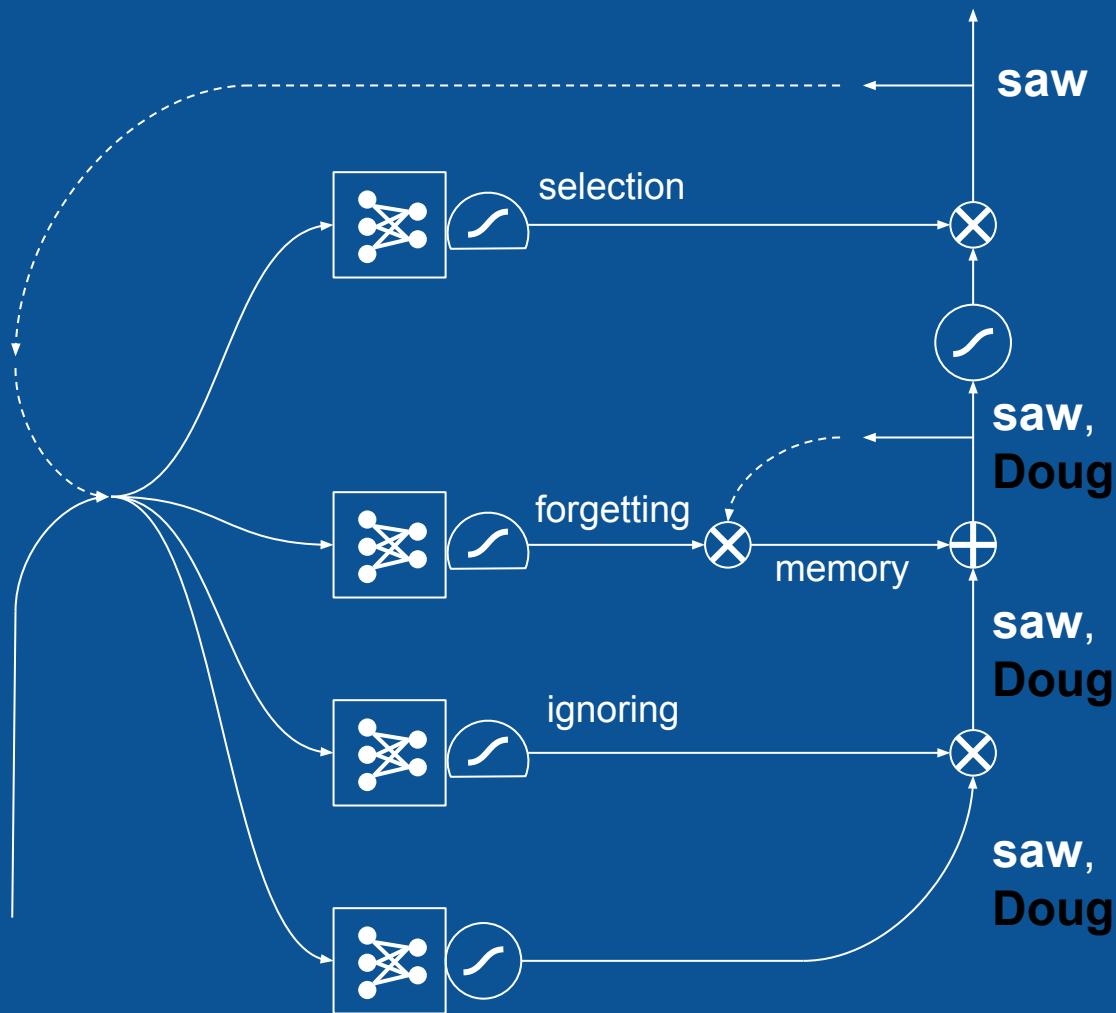
Jane saw Spot.
Doug ...



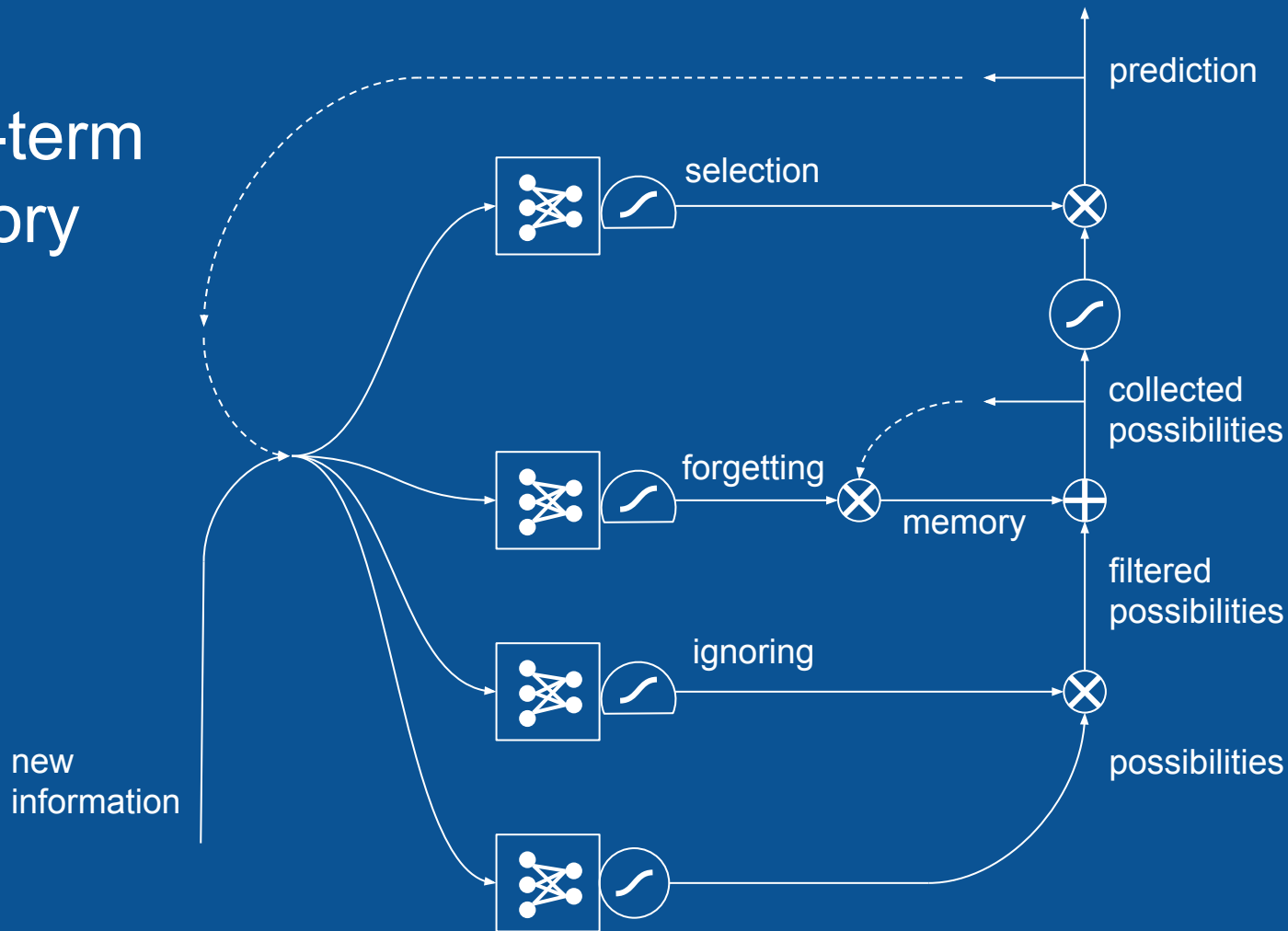
long
short-term
memory

Doug,
Jane,
Spot

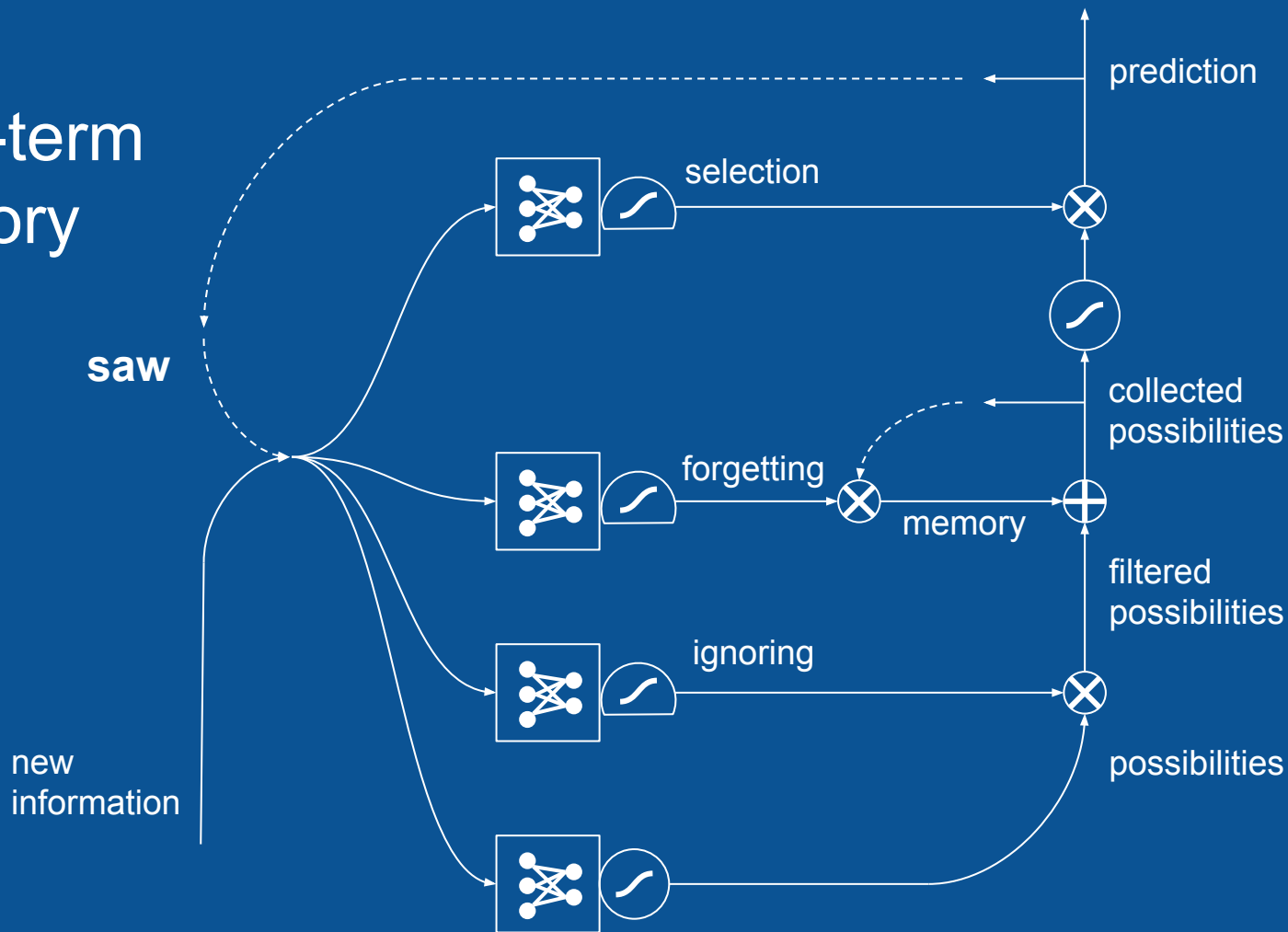
Jane saw Spot.
Doug ...



long short-term memory



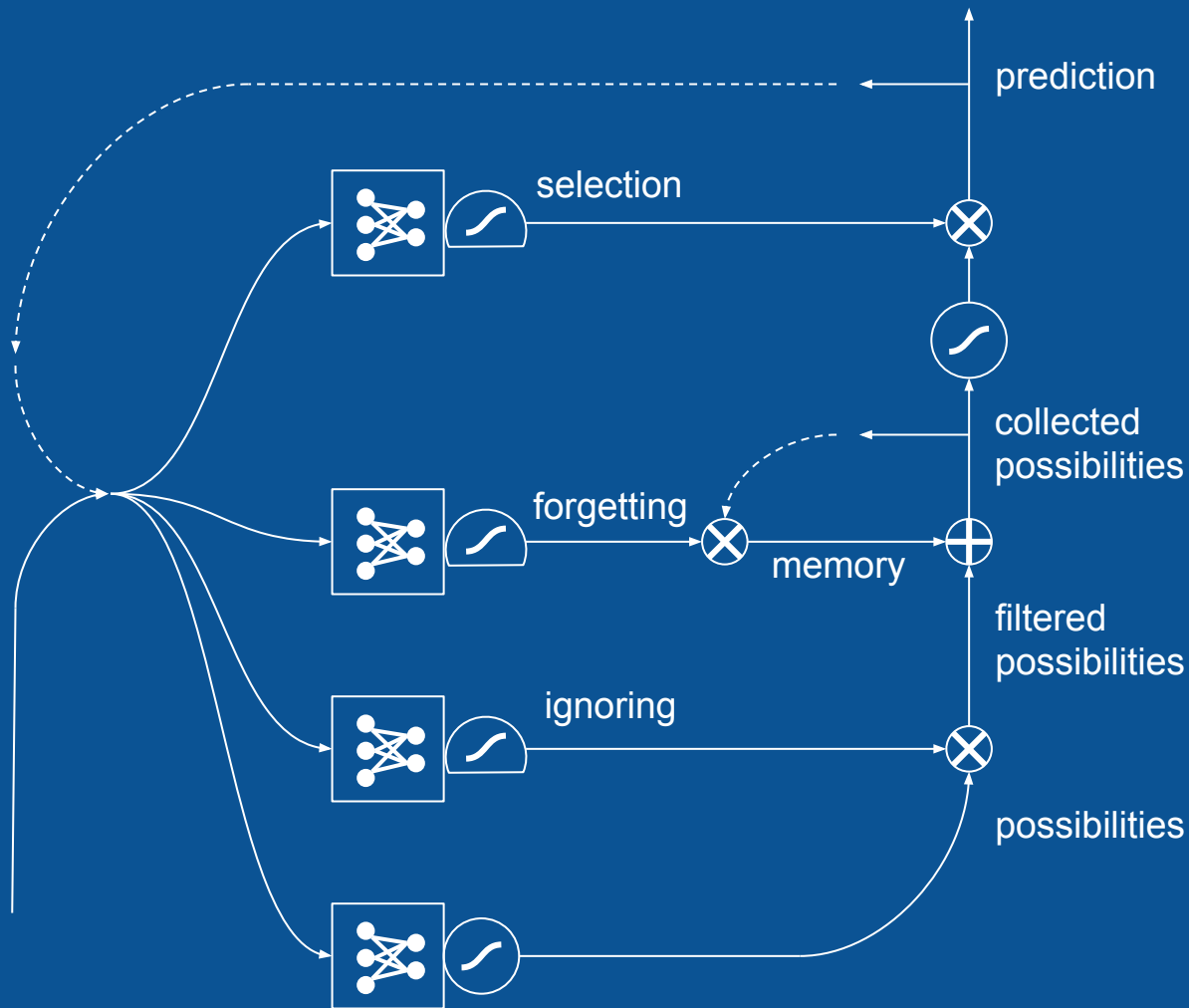
long short-term memory



long short-term memory

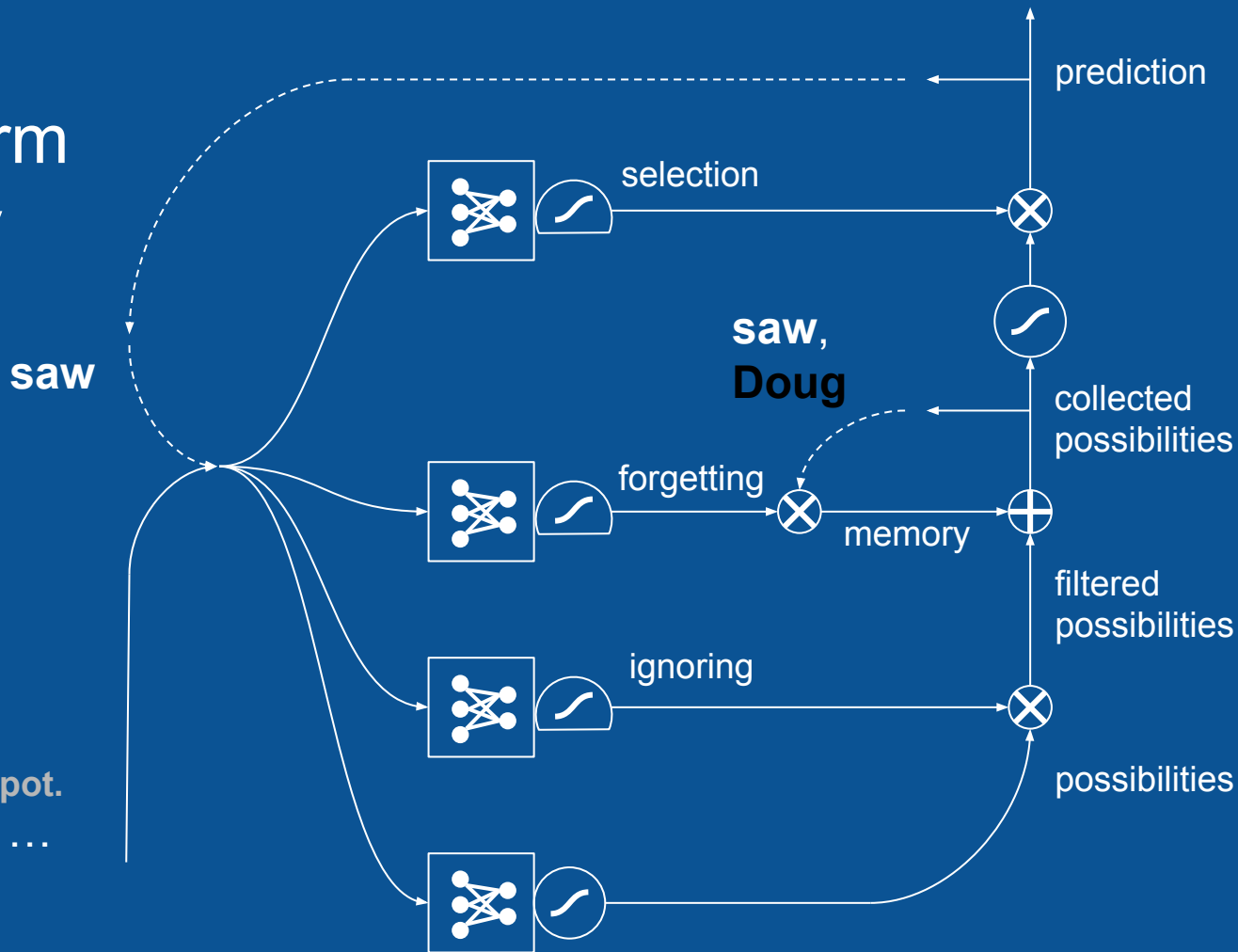
saw

Jane saw Spot.
Doug **saw** ...



long short-term memory

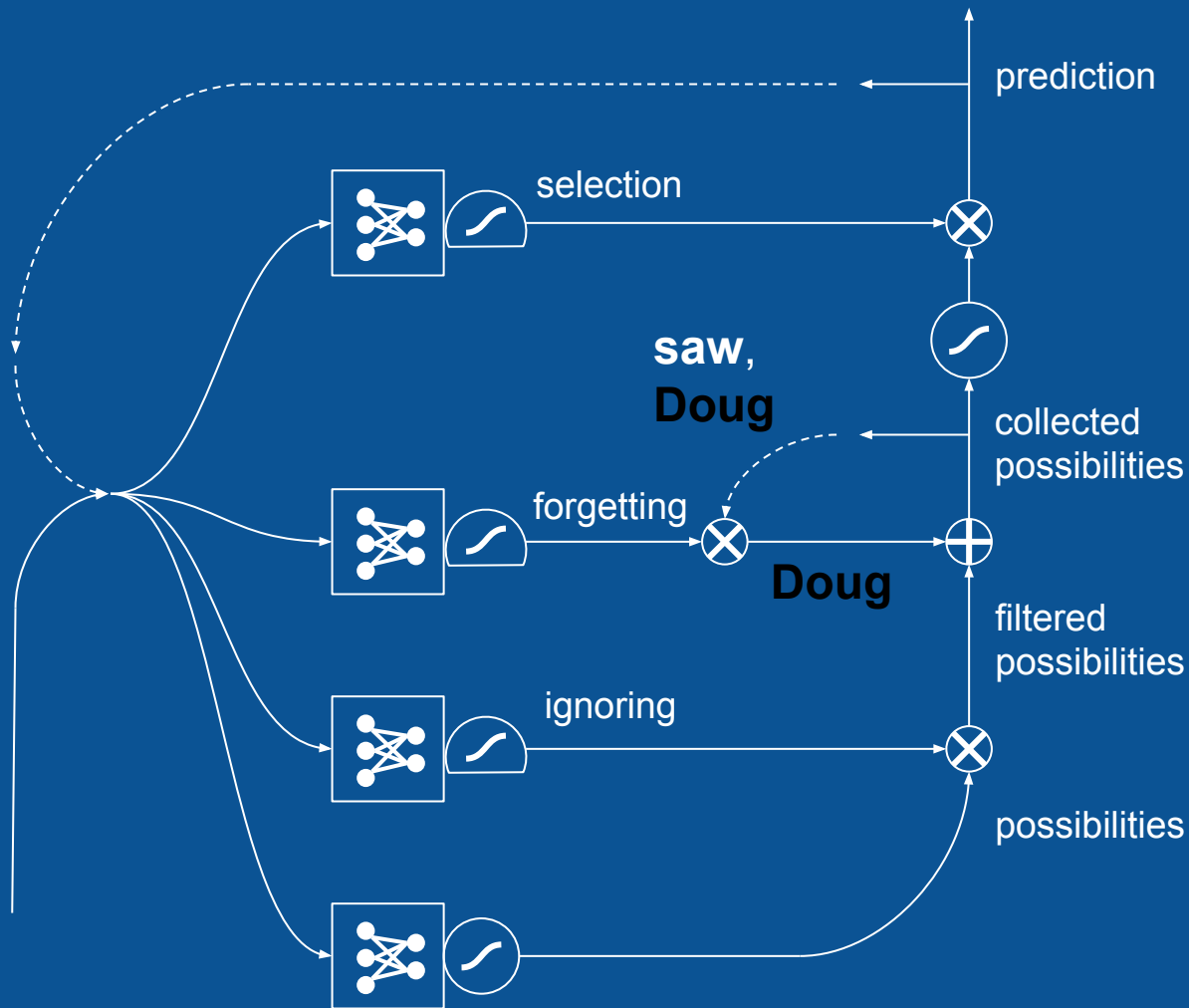
Jane saw Spot.
Doug **saw** ...



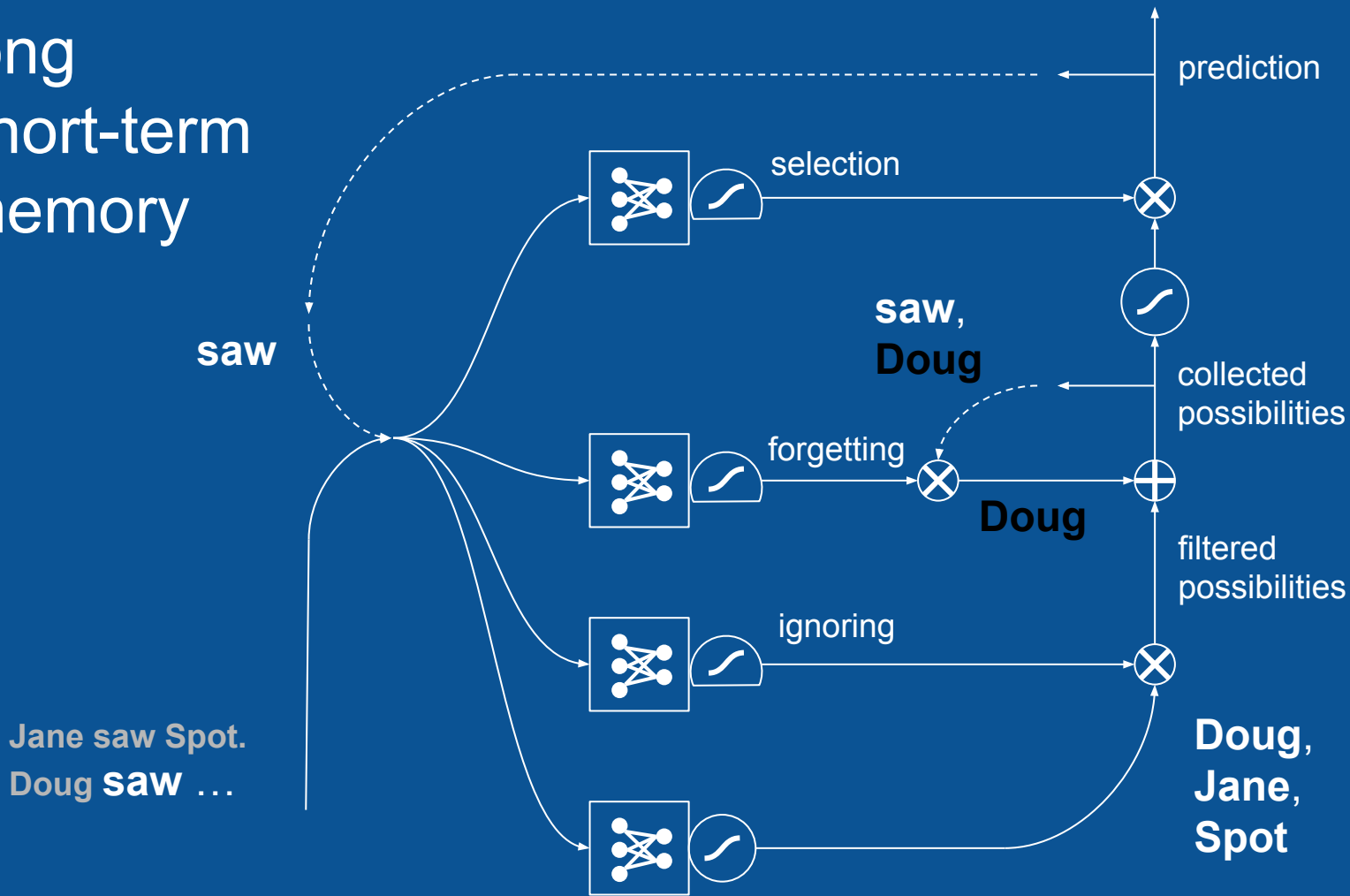
long
short-term
memory

saw

Jane saw Spot.
Doug **saw** ...



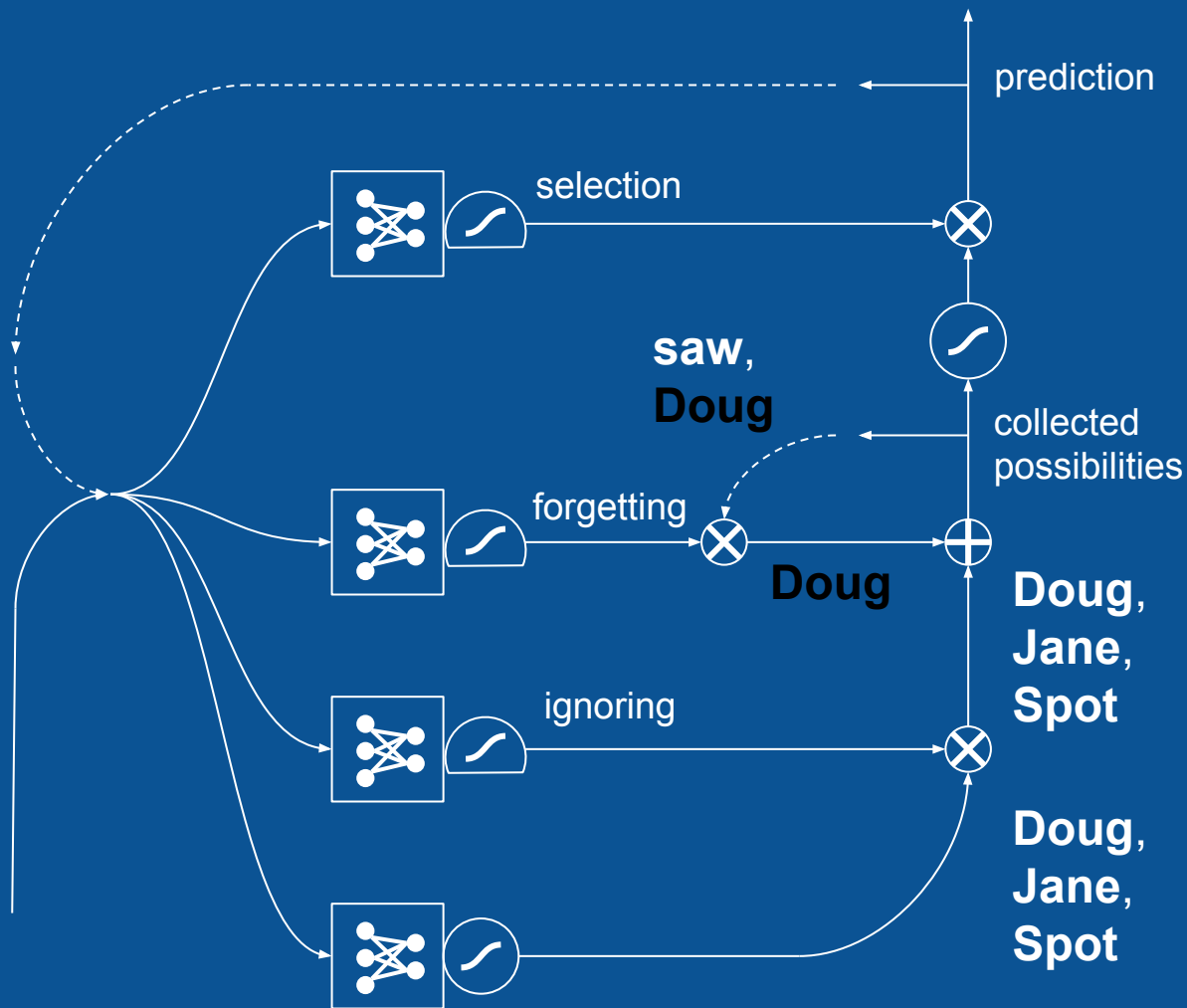
long
short-term
memory



long
short-term
memory

Jane saw Spot.
Doug **saw** ...

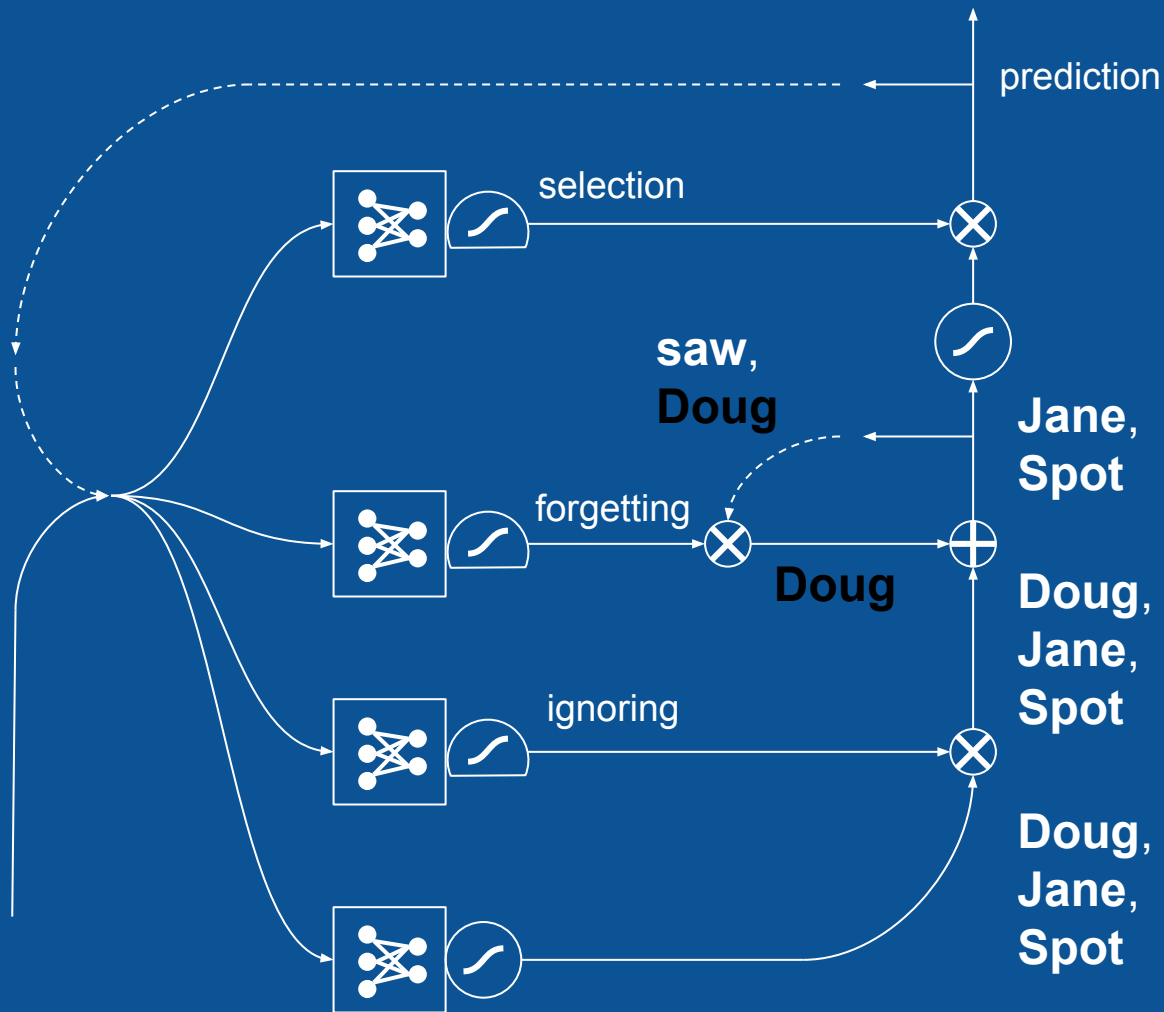
saw



long
short-term
memory

Jane saw Spot.
Doug **saw** ...

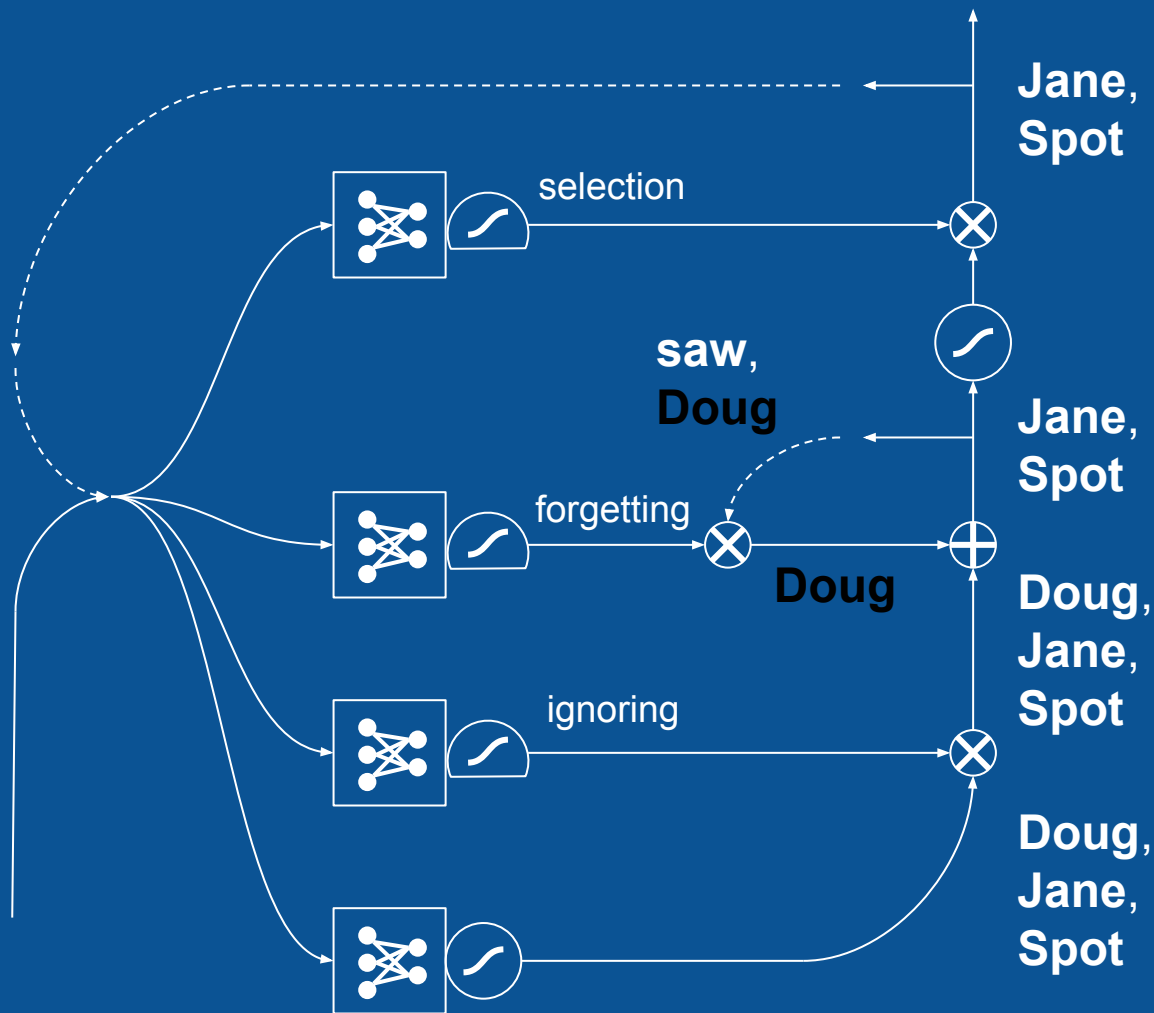
saw



long
short-term
memory

saw

Jane saw Spot.
Doug **saw** ...



Traditional LSTM with forget gates.^{[2][3]}

Initial values: $c_0 = 0$ and $h_0 = 0$. The operator \circ denotes the **Hadamard product** (entry-wise product).

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$

$$h_t = o_t \circ \sigma_h(c_t)$$

Variables

- x_t : input vector
- h_t : output vector
- c_t : cell state vector
- W , U and b : parameter matrices and vector
- f_t , i_t and o_t : gate vectors
 - f_t : Forget gate vector. Weight of remembering old information.
 - i_t : Input gate vector. Weight of acquiring new information.
 - o_t : Output gate vector. Output candidate.

Activation functions

- σ_g : The original is a **sigmoid function**.
- σ_c : The original is a **hyperbolic tangent**.
- σ_h : The original is a hyperbolic tangent, but the peephole LSTM paper suggests $\sigma_h(x) = x$.^{[18][19]}

source:
Wikipedia

Resources

Brandon Rohrer's [blog on RNNs and LSTMs](#)

Chris Olah's [tutorial](#)

Andrej Karpathy's

[Blog post](#)

[RNN code](#)

[Stanford CS231n lecture](#)

The [DeepLearning 4J](#) tutorial has some helpful discussion and a longer list of good resources.

[How Neural Networks Work](#) [\[video\]](#)

Credits (all images CC0)

[Pizza image](#)

[Sushi image](#)

[Waffles image](#)

Appendix

A vector is a list of values

“Tonight I think
we’re going to
have sushi.”

=

Pizza

0

Sushi

1

Waffles

0

=

Dinner prediction vector

0

1

0

A vector is a list of values

“High is 67 F.
Low is 43 F.
Wind is 13 mph.
.25 inches of rain.
Relative humidity
is 83%.”



High
temperature

67

Low
temperature

43

Wind speed

13

Precipitation

.25

Humidity

.83



Weather vector

67

43

13

.25

.83

A vector is a list of values

