Deep Learning (Parte 2)

Índice Geral

1 Introdução

- Para compreender devidamente redes neurais recorrentes, é necessário dominar conceitos fundamentais associados a sistemas dinâmicos.
- O conceito mais básico é o de **estado do sistema**, que computacionalmente é implementado na forma de uma memória interna.
- O estado pode ser representado na forma de um vetor de variáveis, capazes de caracterizar completamente a configuração atual do sistema. Mais do que isso: com o conhecimento do estado do sistema no instante presente, de sua dinâmica e das entradas que atuam sobre o sistema do instante presente em diante, o que vai ocorrer com o sistema dinâmico do instante presente em diante não depende do que ocorreu com o sistema no passado, pois tudo está condensado nas variáveis de estado do sistema.
- Logo, as variáveis de estado, por definição, integram tudo o que aconteceu com o sistema no passado.

- Definição formal: O estado de um sistema é formado por um conjunto de valores de grandezas físicas, necessário e suficiente para caracterizar a situação física deste sistema.

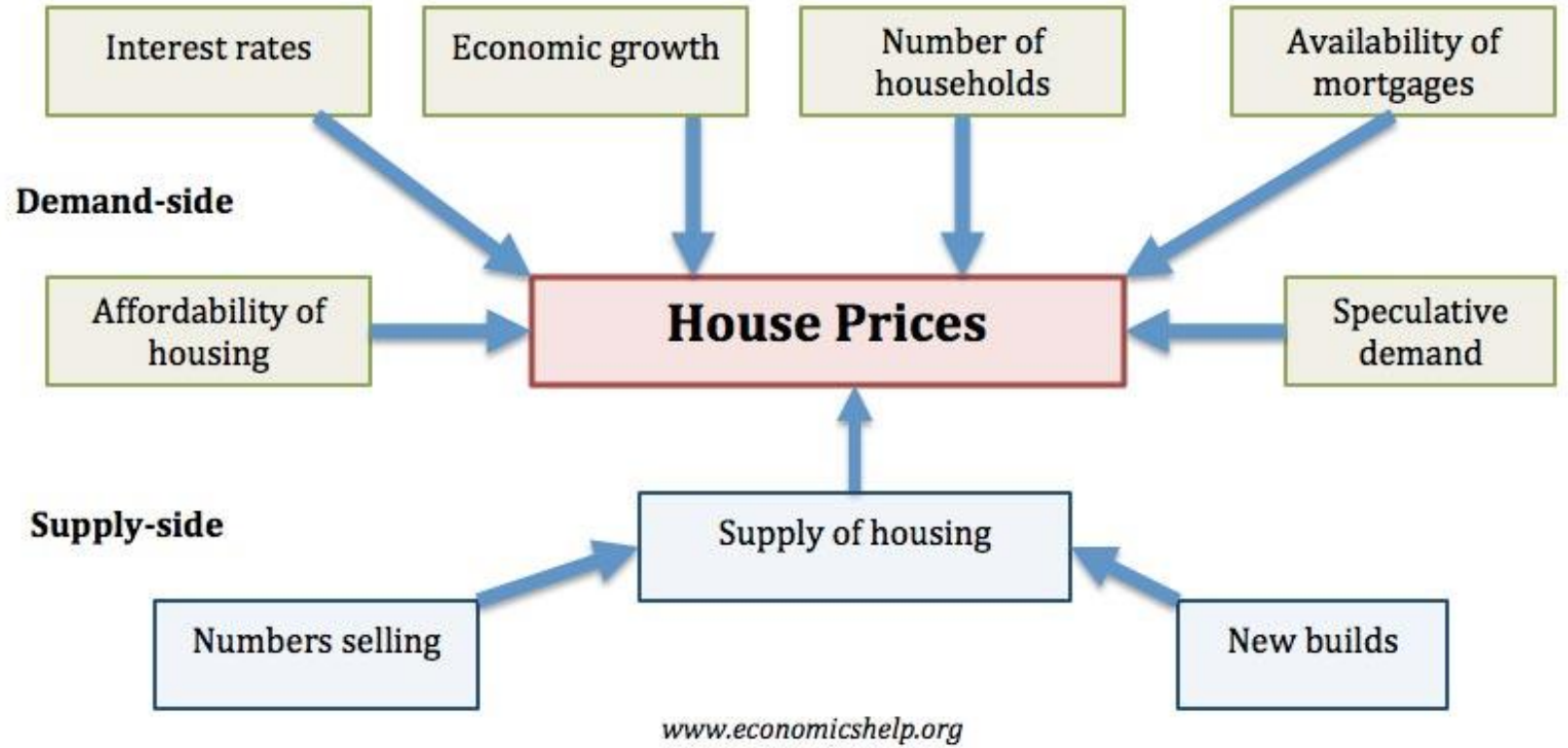


Figura 1 – Exemplo ilustrativo de variáveis temporais que podem definir completamente a dinâmica do preço de um imóvel residencial

- Matematicamente, o estado de um sistema dinâmico corresponde a um ponto no **espaço de estados**. Ou seja, cada coordenada está associada a uma variável de estado. Dependendo da cardinalidade do conjunto (número de variáveis de estados), o espaço de estados pode ser de dimensão finita (uma ou mais) ou infinita (no caso de fluido turbulento, por exemplo).
- O estado pode ser contínuo ou discreto.
- Os sistemas dinâmicos físicos processam matéria, energia e/ou informação. A evolução do seu estado no tempo é causal ou não-antecipativa.
- A **dinâmica** do sistema é a lei de variação do estado no tempo. Ela pode ser contínua ou discreta. A dinâmica contínua indica a direção de variação do estado, enquanto que a dinâmica discreta indica o próximo estado.
- A **trajetória** descreve a evolução no tempo do estado do sistema. Em outras palavras, descreve o deslocamento de um ponto no espaço de estados regido pela dinâmica do sistema. A trajetória é parametrizada no tempo.

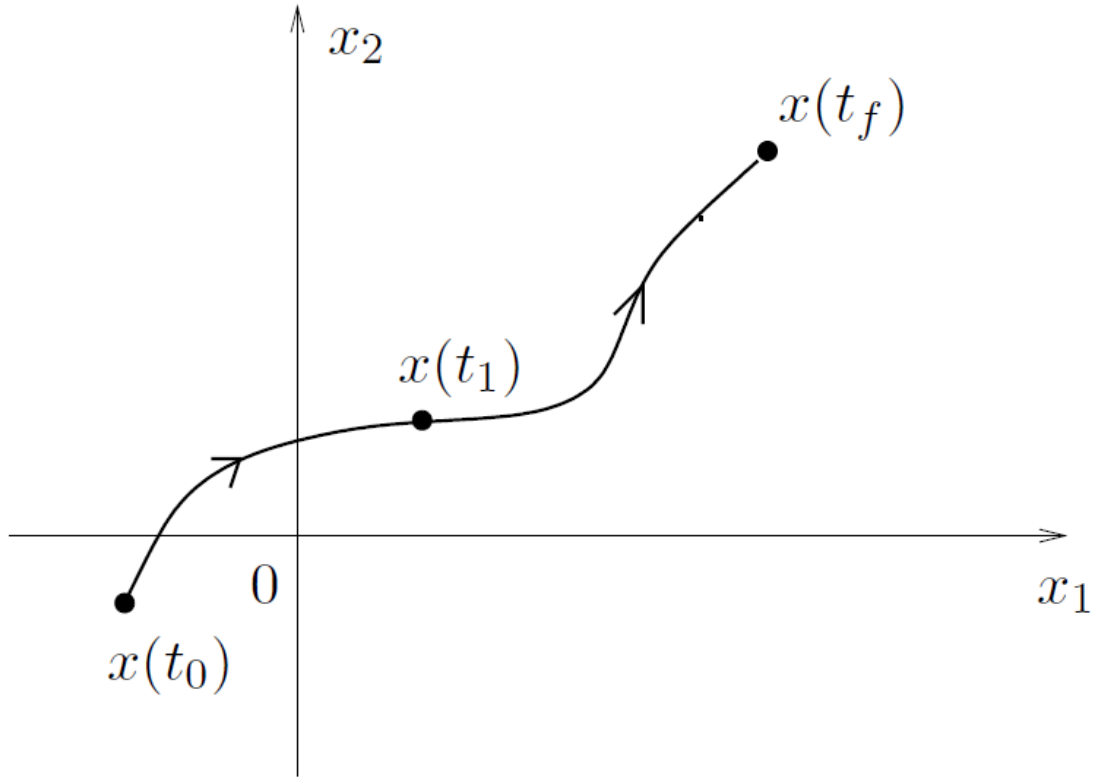


Figura 2 – Trajetória no espaço de estados para dinâmica contínua e estados contínuos

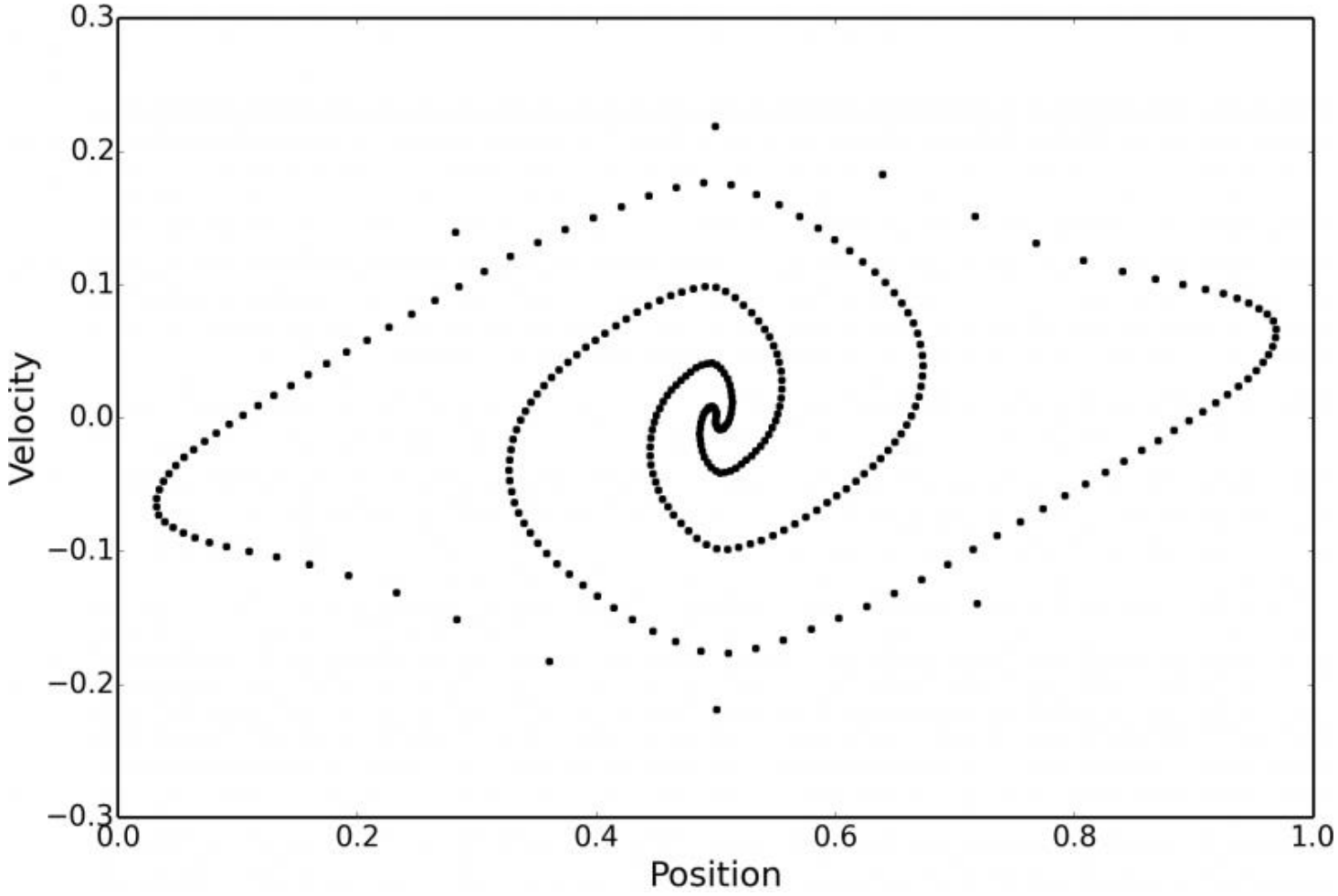


Figura 3 – Duas trajetórias no espaço de estados (partindo de condições iniciais distintas) para dinâmica discreta e estados contínuos

1.1 Taxonomia de sistemas dinâmicos

		ESPAÇO DE ESTADOS	
		contínuo	discreto
DINÂMICA	contínua	admite uma descrição matemática na forma de um sistema de equações diferenciais	vidros de spin
	discreta	admite uma descrição matemática na forma de um sistema de equações a diferenças	autômato

- A dinâmica pode ser estacionária (sistemas invariantes no tempo: quem não varia é a dinâmica, pois o estado pode variar no tempo) ou não-estacionária (sistemas variantes no tempo).

2 *Backpropagation Through Time* (BPTT)

- Para redes neurais recorrentes com realimentação de saída, é possível implementar uma técnica baseada em gradiente para ajuste dos pesos sinápticos.
- A ideia é "desdobrar" a rede neural recorrente ao longo do tempo, produzindo arquiteturas como na Figura 4 a seguir.

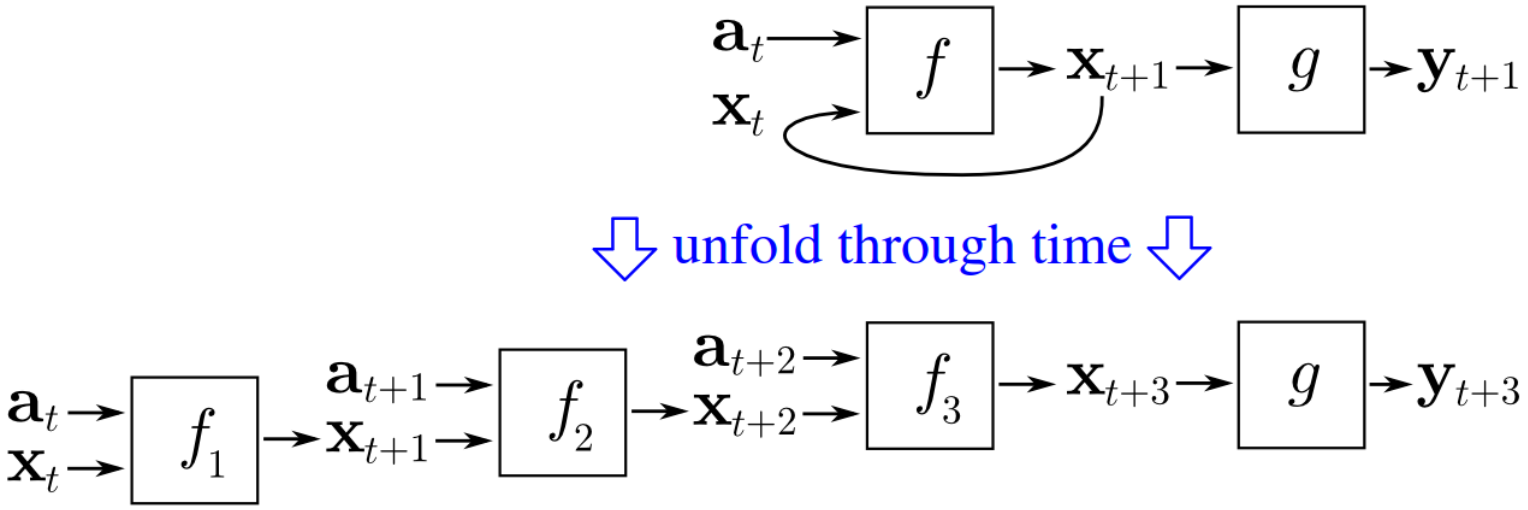
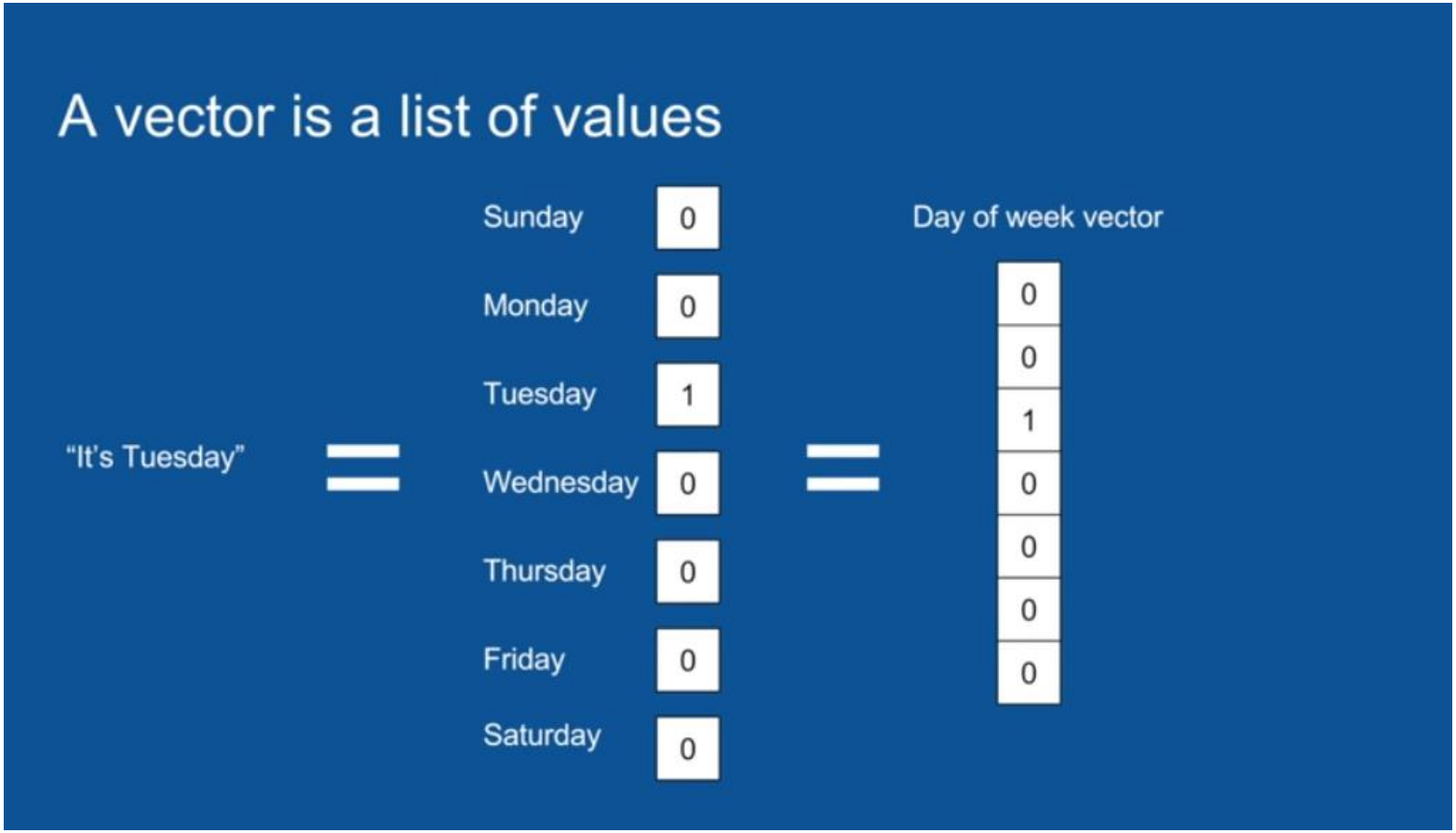


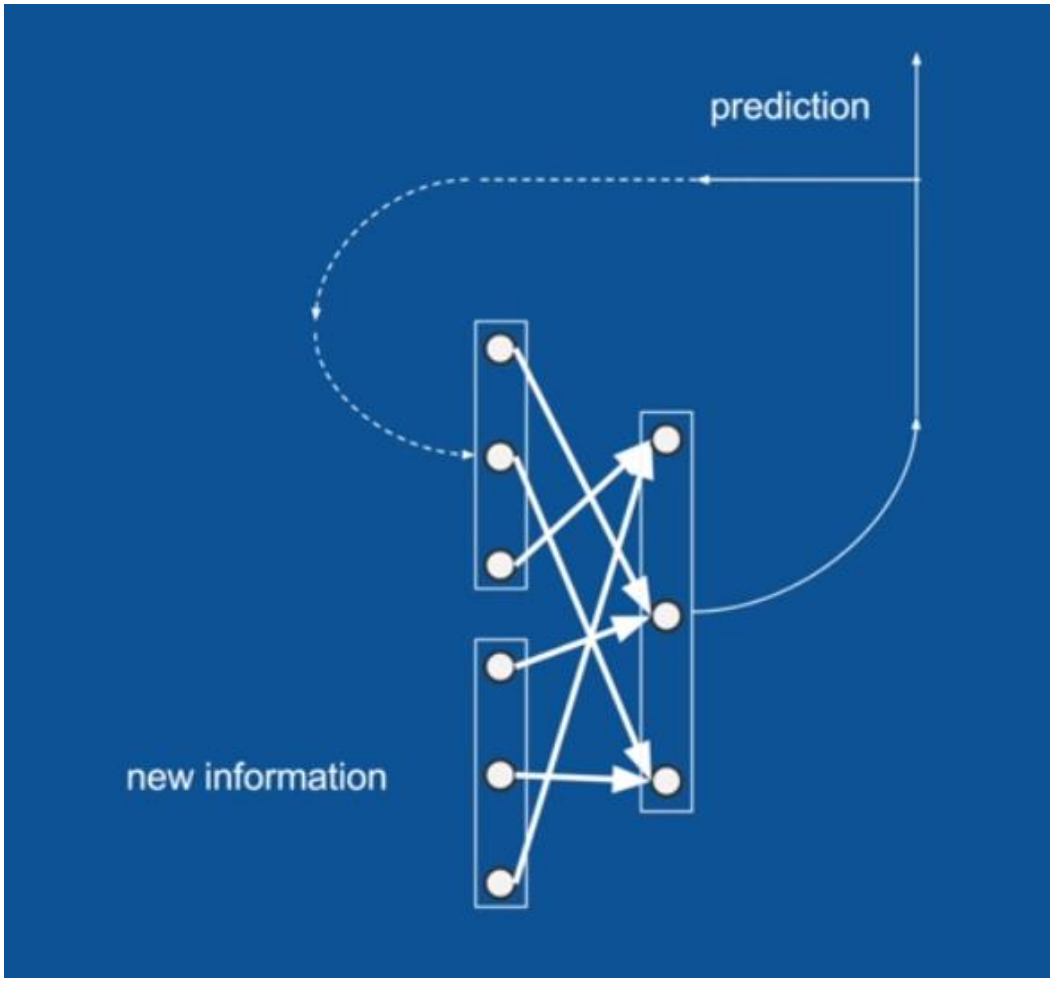
Figura 4 – Representação do desdobramento no tempo, para posterior aplicação do *backpropagation through time* (BPTT) (Fonte: http://en.wikipedia.org/wiki/Backpropagation_through_time).

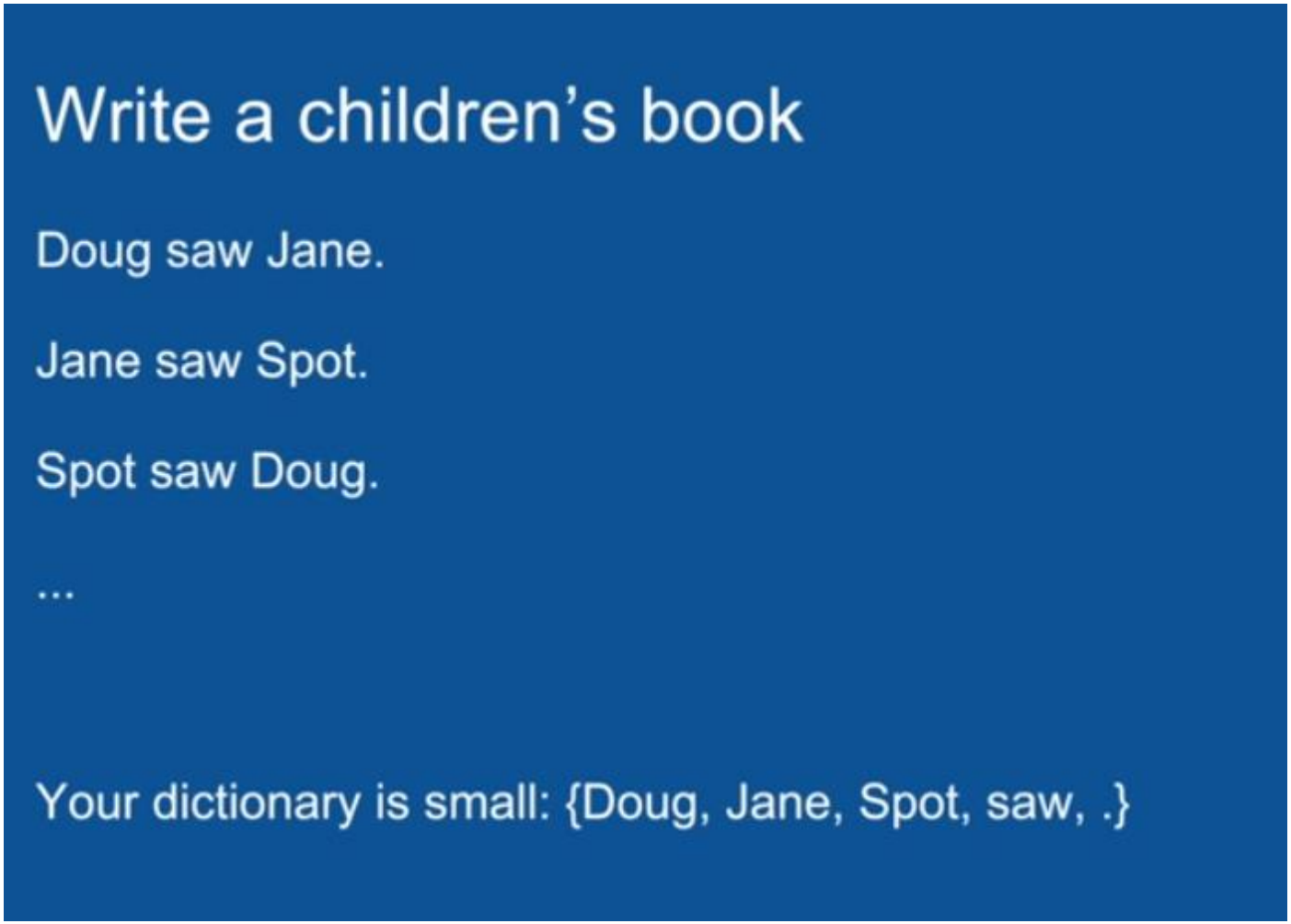
- Geralmente, uma etapa de BPTT é aplicada a cada apresentação de uma amostra de treinamento, ou seja, o treinamento é *on-line*.
- Os pesos associados ao mapeamento *f* são atualizados como a média dos pesos após a etapa de BPTT, tomando as *k* instâncias de *f*, sendo que no caso da Figura 4 tem-se $k = 3$.
- BPTT tende a ser mais eficiente para ajuste de pesos de redes neurais recorrentes do que buscas heurísticas, como técnicas evolutivas (MCDONNELL & WAGEN, 1994; SJÖBERG *et al*., 1995). No entanto, os desafios associados à existência de mínimos locais são mais acentuados do que no caso de redes neurais não-recorrentes.
- Para mais detalhes referentes ao formalismo envolvido, consultar NERRAND *et al.* (1994), PEARLMUTTER (1995), PINEDA (1987), PINEDA (1989), WERBOS (1990) e WILLIAMS & ZIPSER (1989).

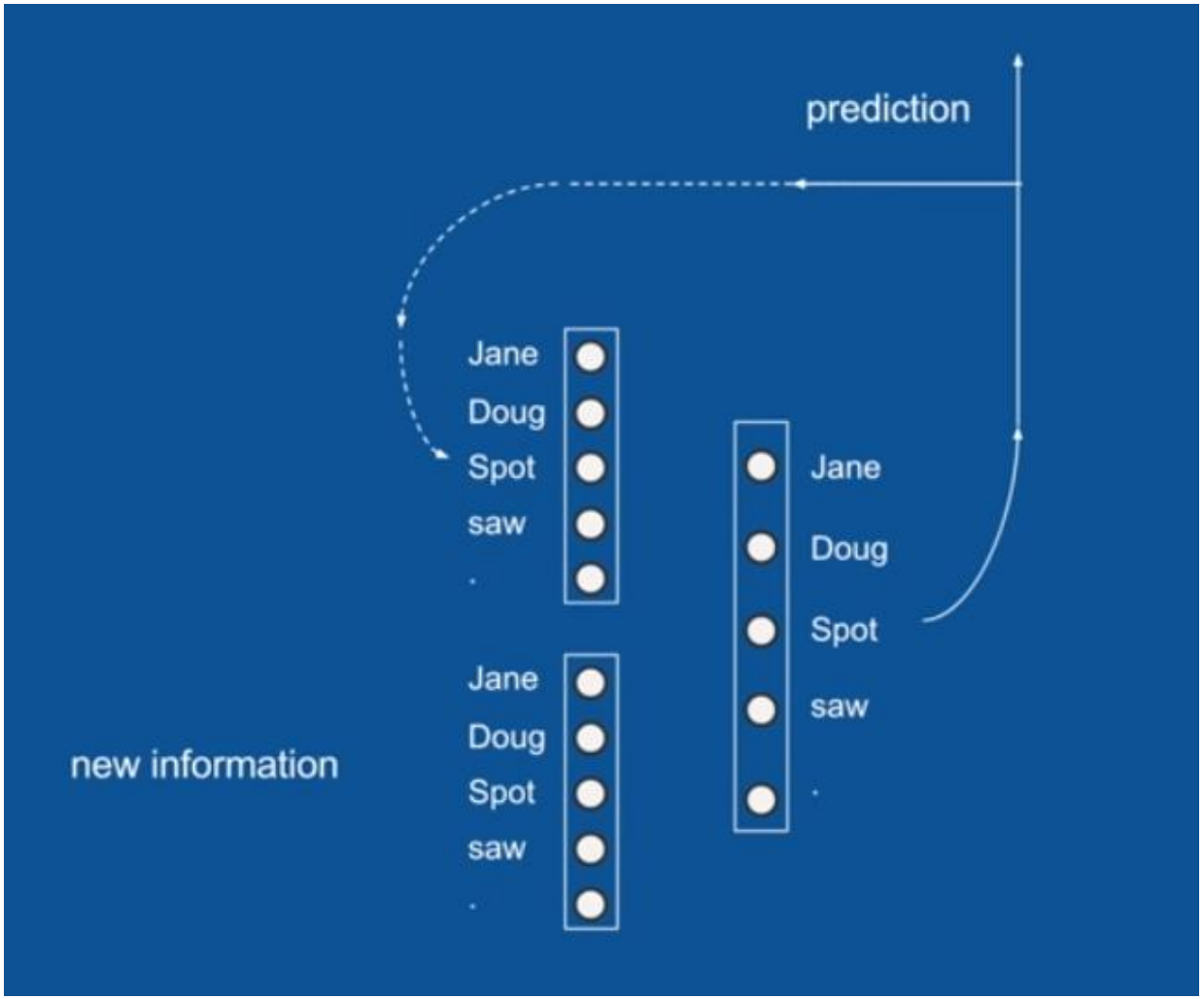
3 Motivação para o bloco LSTM

- O bloco LSTM (*Long Short Term Memory*) foi proposto em 1997 (HOCHREITER & SCHMIDHUBER, 1997) e, como o próprio nome indica, foi concebido para viabilizar a implementação conjunta de memórias de longo e de curto prazos em redes neurais recorrentes.
- Sugere-se a leitura do paper de revisão com as principais conquistas na área: GREFF et al. (2017).
- Esta seção está baseada no vídeo de Brandon Rohrer (cientista de dados sênior do Facebook): https://www.youtube.com/watch?v=WCUNPb-5EYI

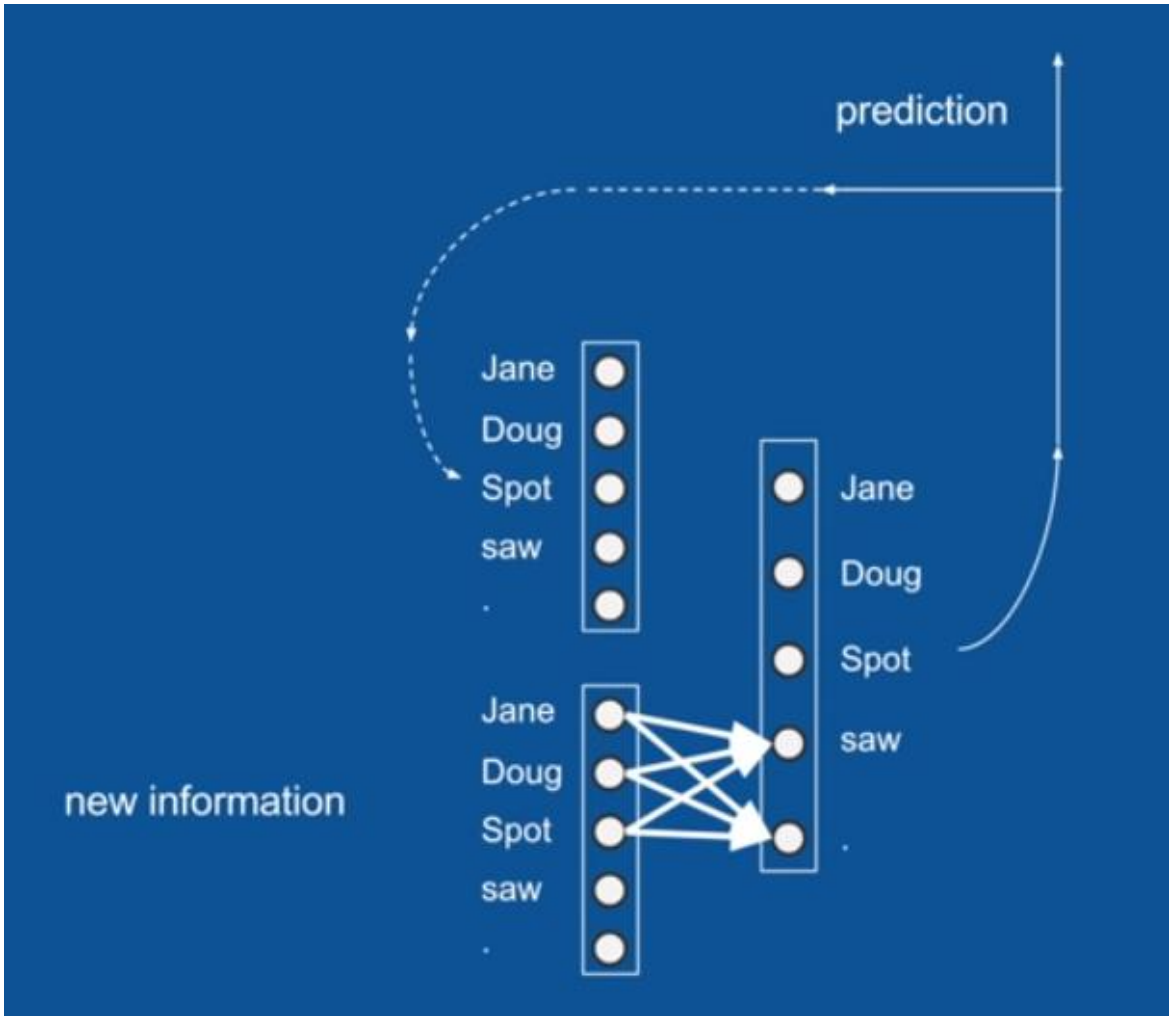

A vector is a list of values
"It's Tuesday"
=
Sunday 0
Monday 0
Tuesday 1
Wednesday 0
Thursday 0
Friday 0
Saturday 0
=
Day of week vector
0
0
1
0
0
0
0


prediction
new information

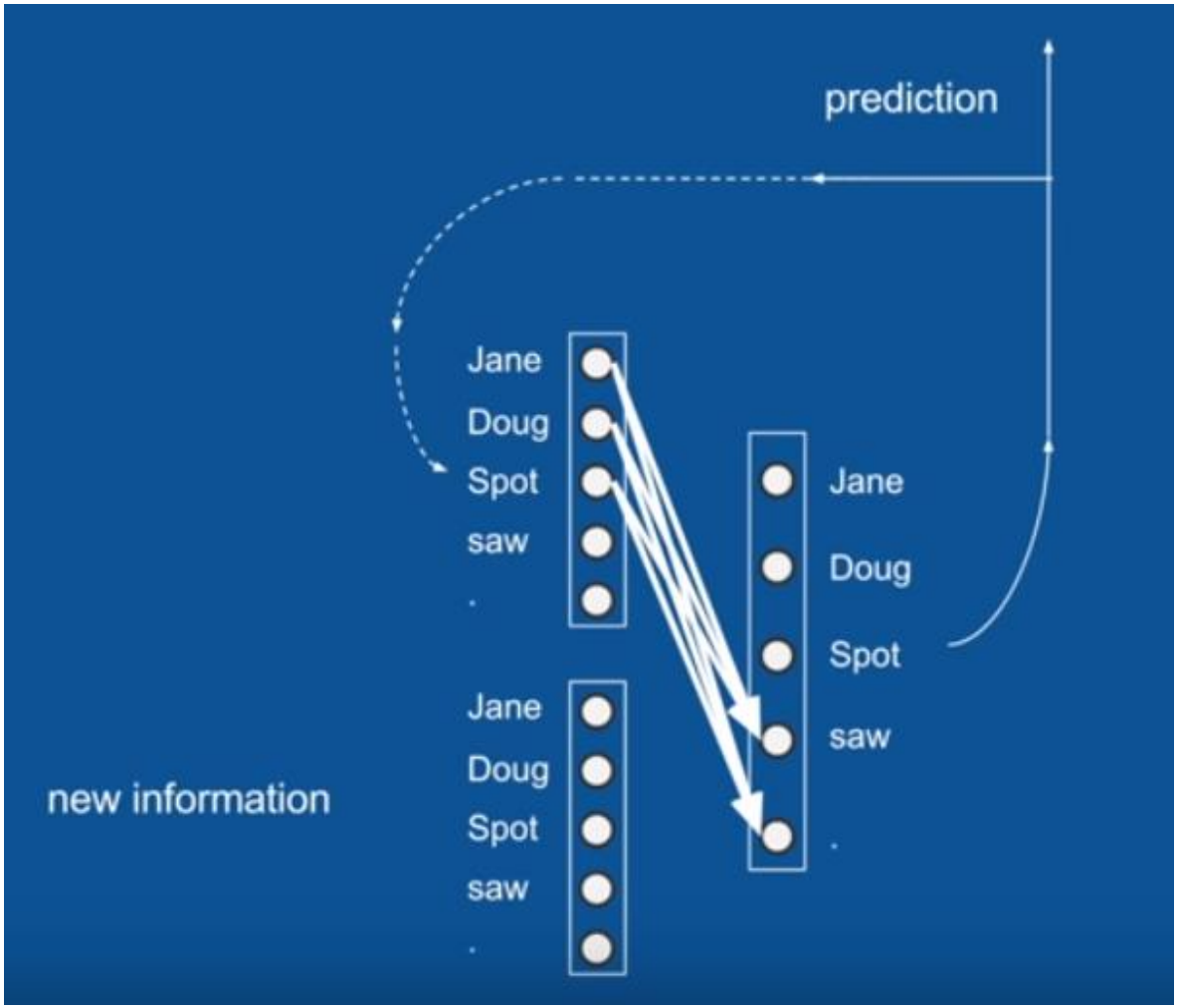

Write a children's book
Doug saw Jane.
Jane saw Spot.
Spot saw Doug.
...
Your dictionary is small: {Doug, Jane, Spot, saw, .}


prediction
Jane
Doug
Spot
saw
.
Jane
Doug
Spot
saw
.
Jane
Doug
Spot
saw
.
new information

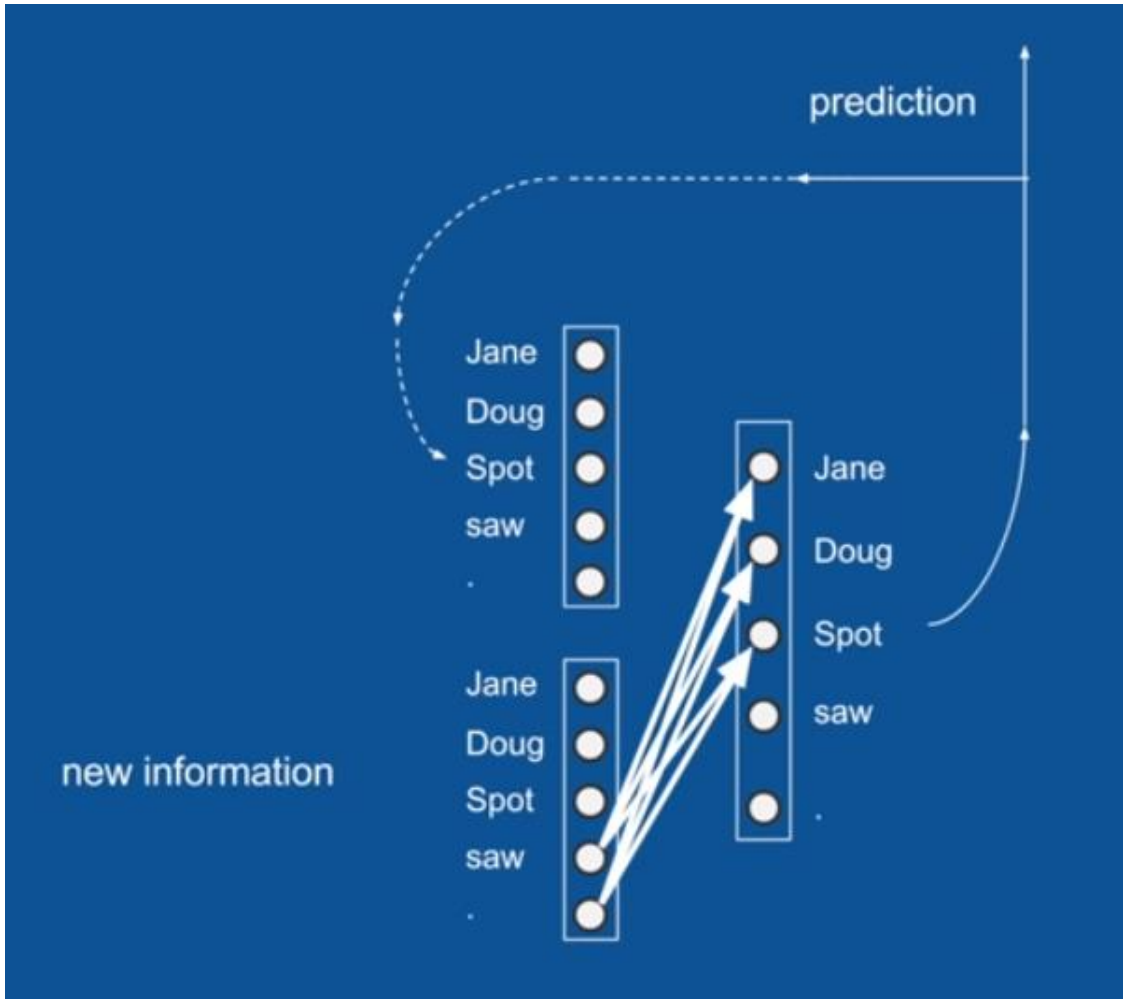
- Após o treinamento, espera-se o surgimento de padrões bem característicos.

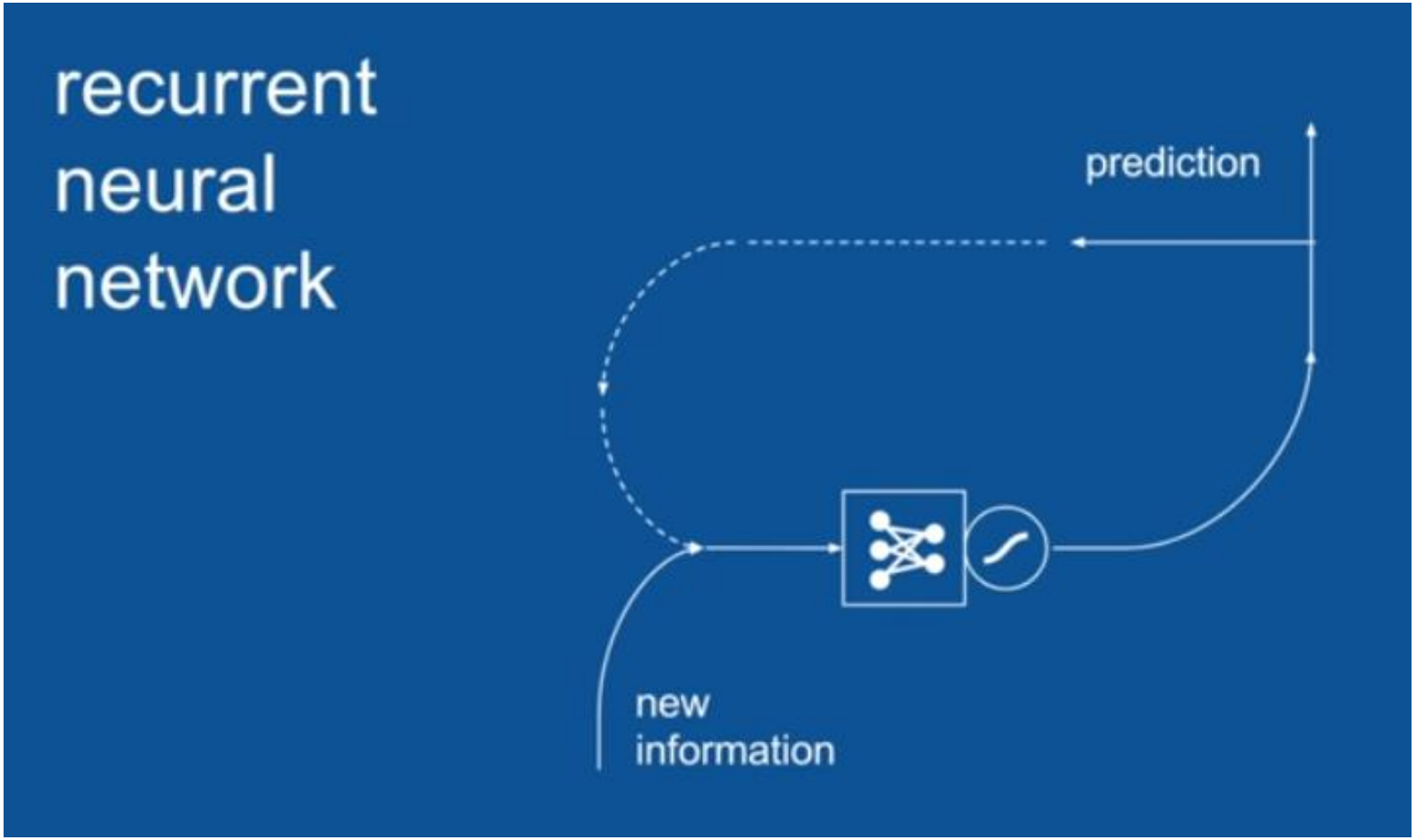


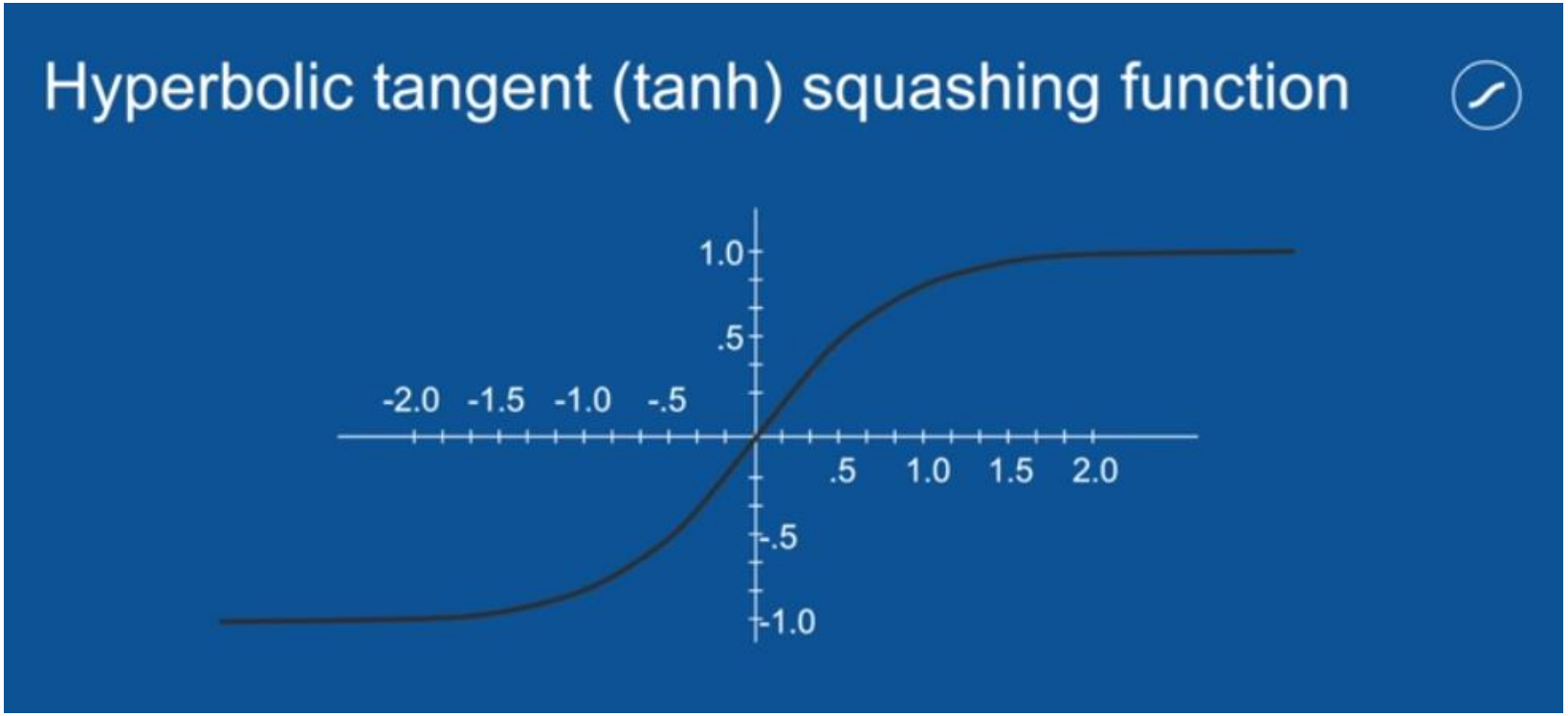
- Após o treinamento, espera-se o surgimento de padrões bem característicos.

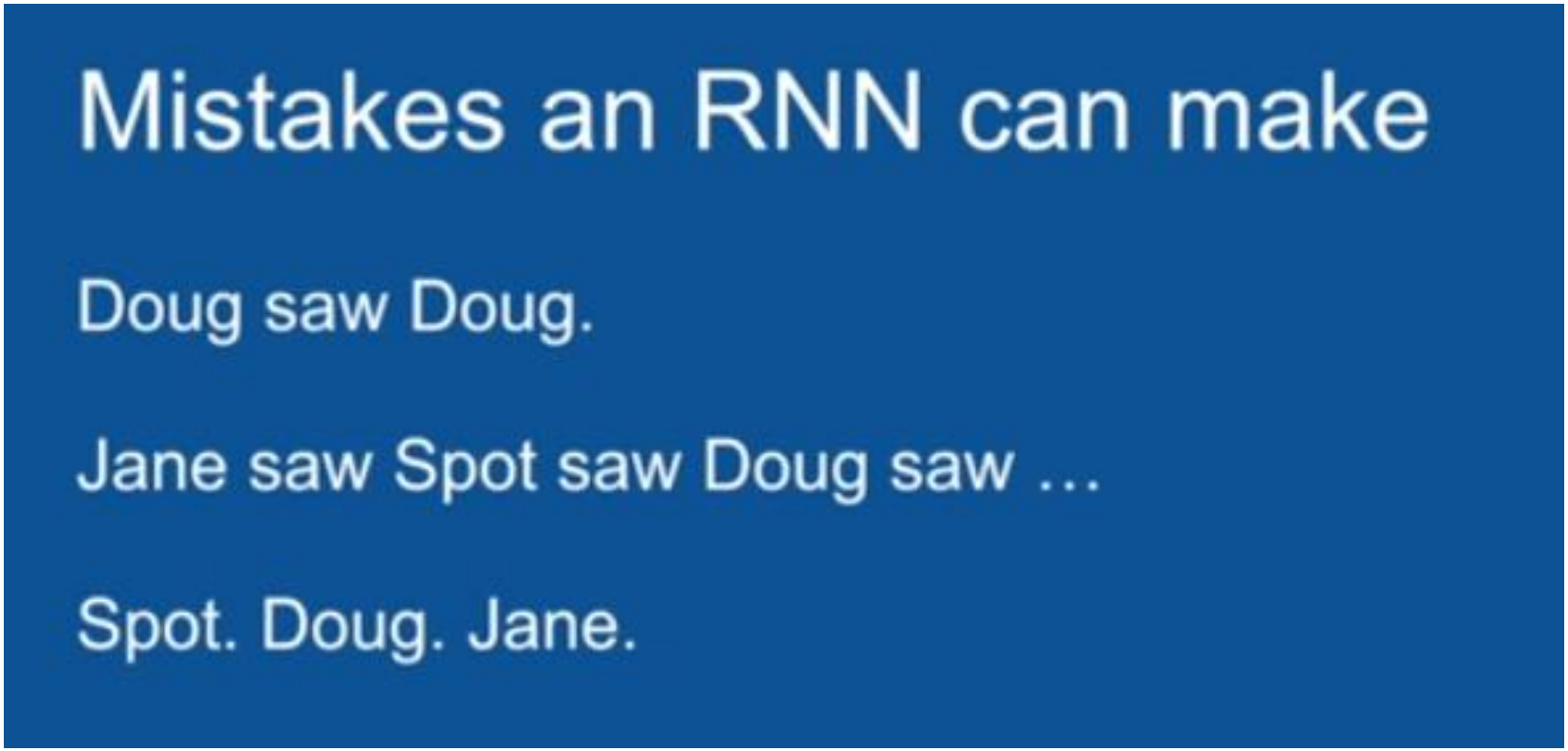


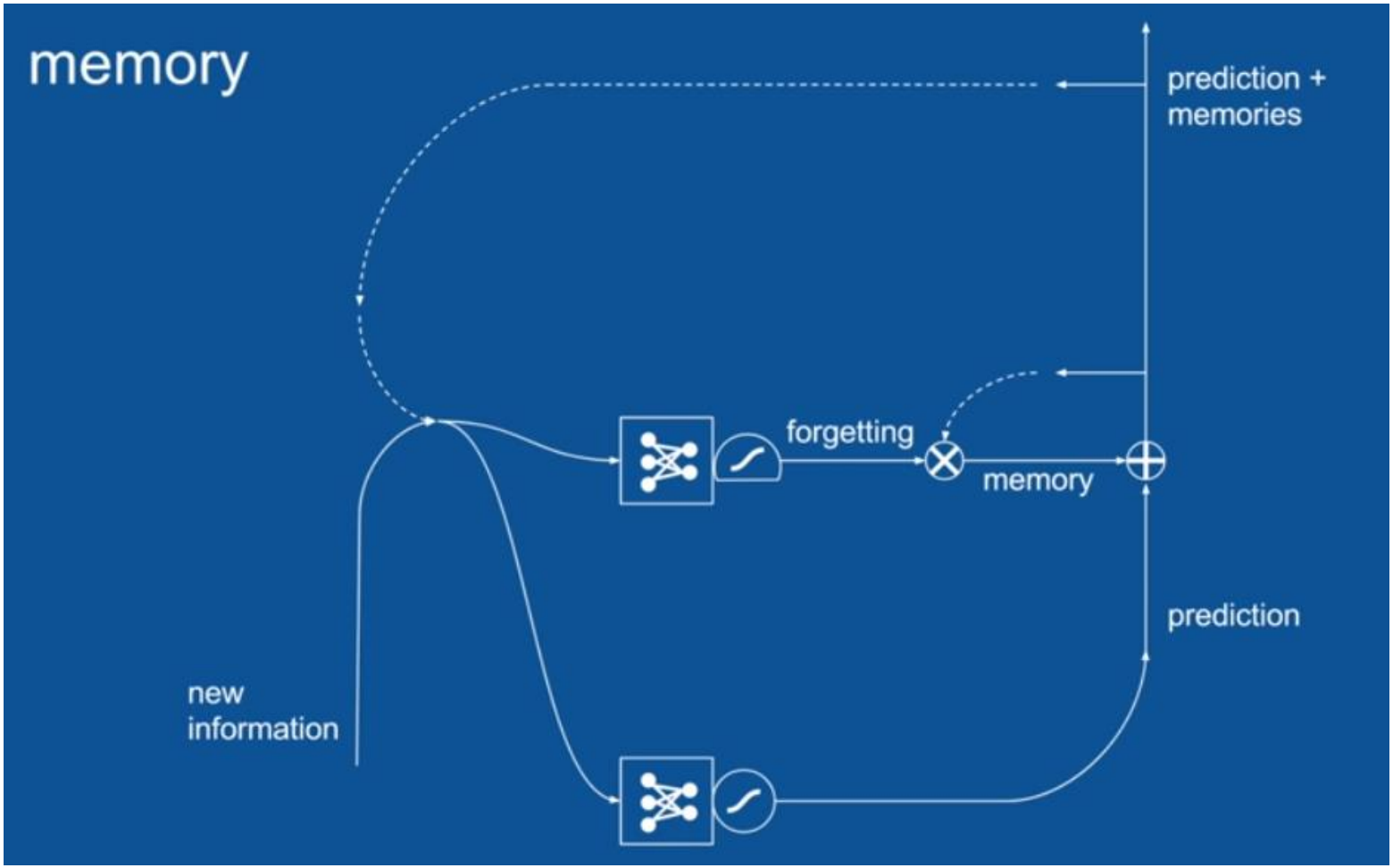
- Após o treinamento, espera-se o surgimento de padrões bem característicos.

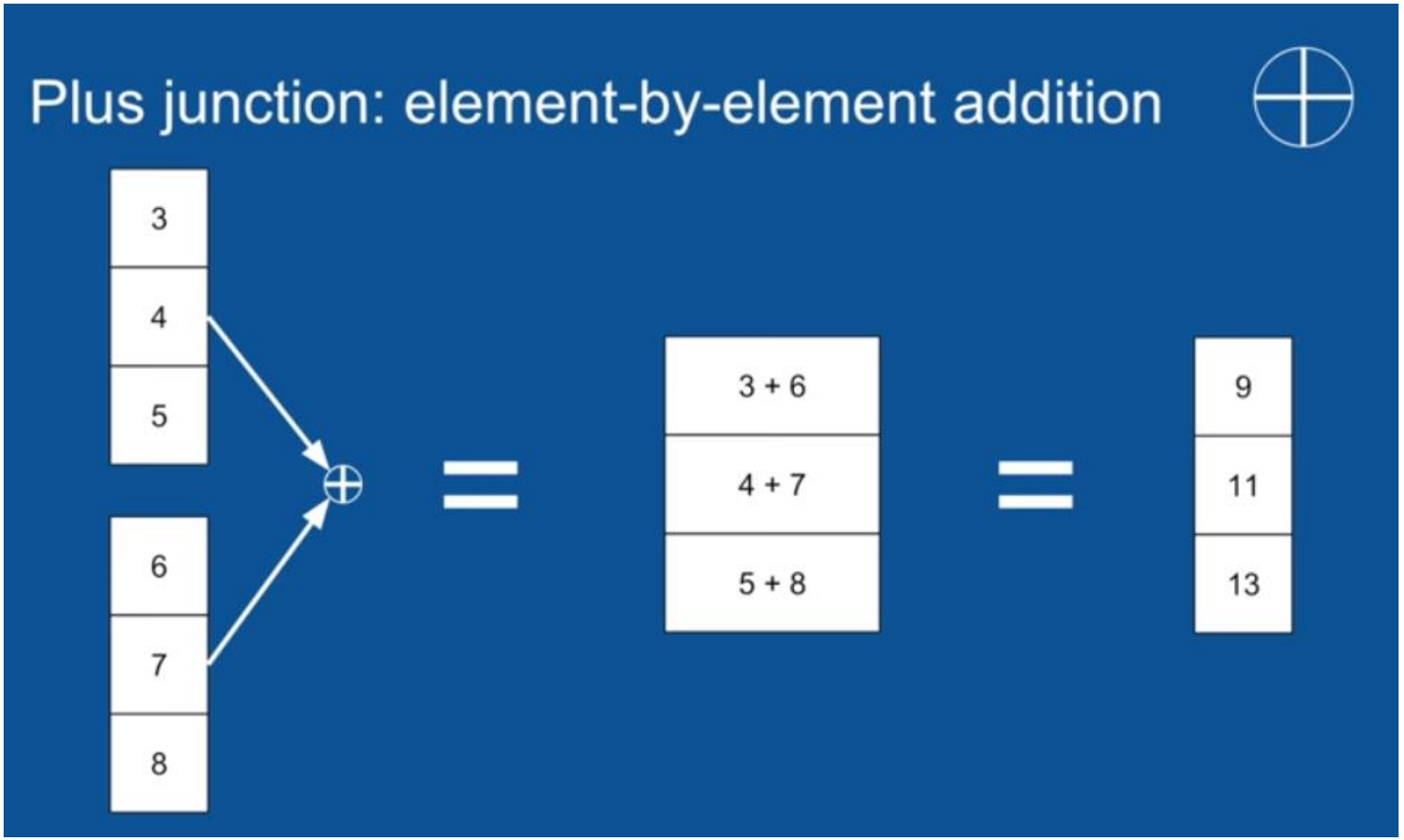


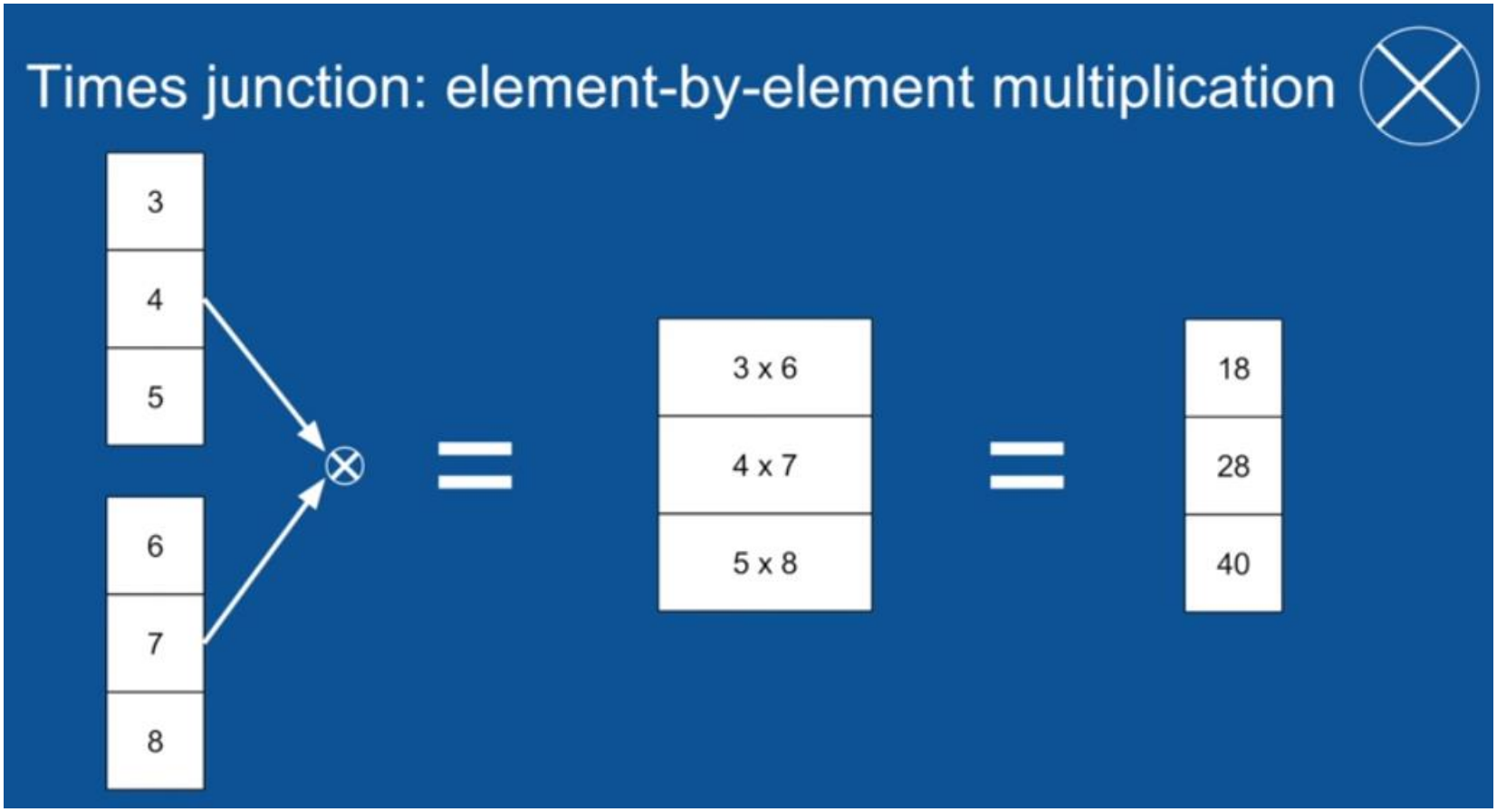

recurrent
neural
network
prediction
new
information

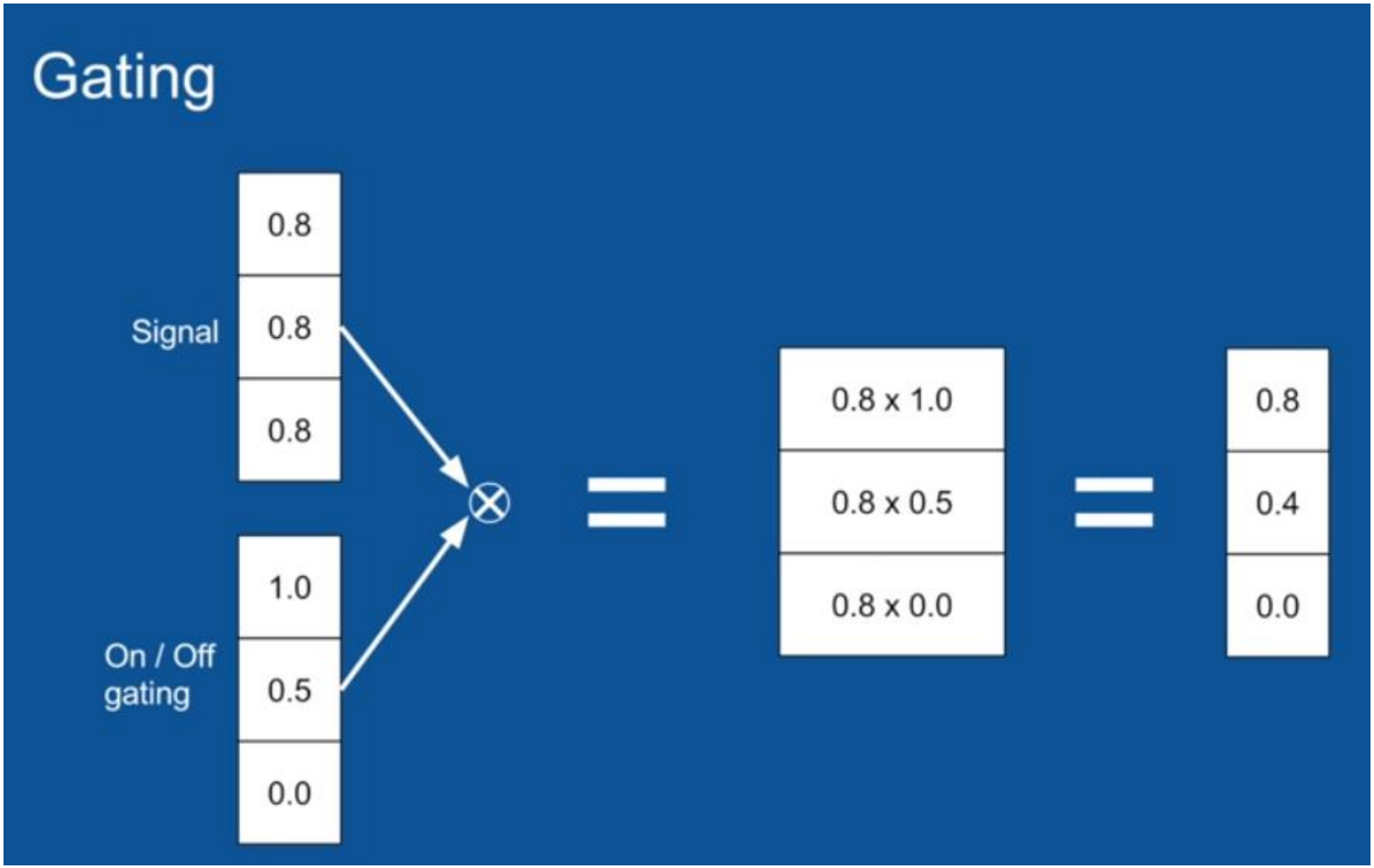

Hyperbolic tangent (tanh) squashing function
1.0
.5
-2.0
-1.5
-1.0
-.5
.5
1.0
1.5
2.0
-.5
-1.0

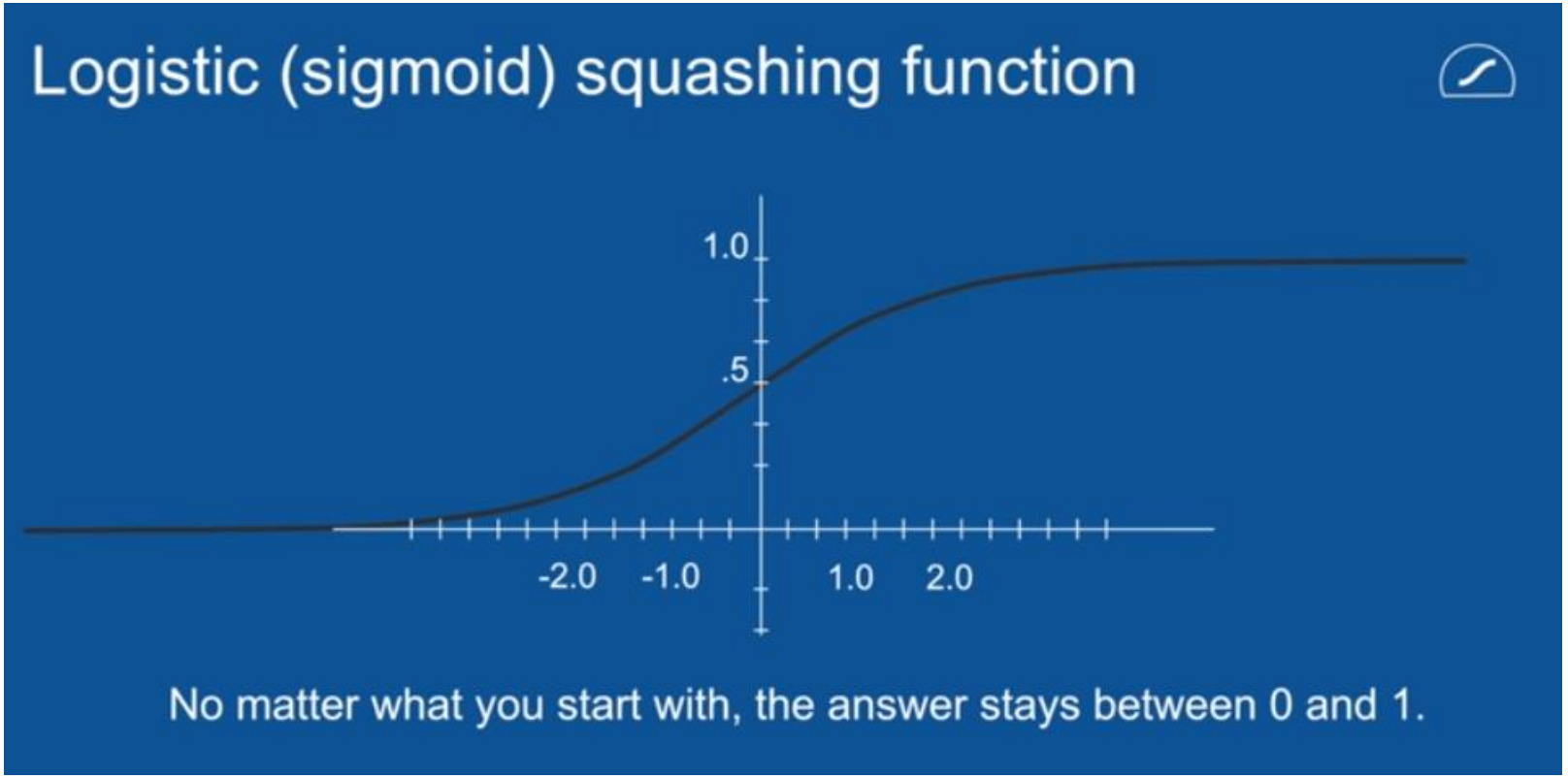

Mistakes an RNN can make
Doug saw Doug.
Jane saw Spot saw Doug saw …
Spot. Doug. Jane.

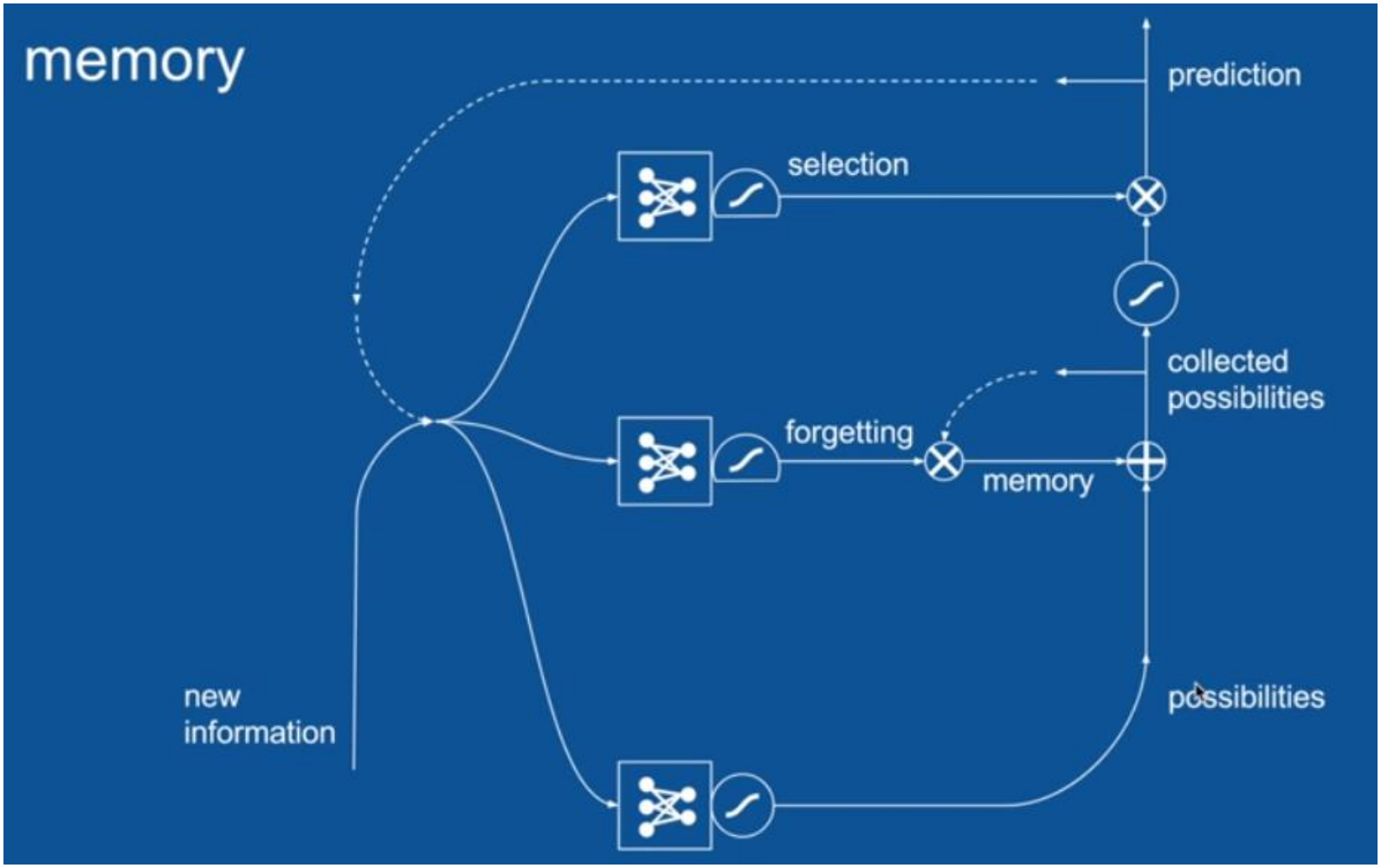

memory
prediction +
memories
forgetting
memory
prediction
new
information

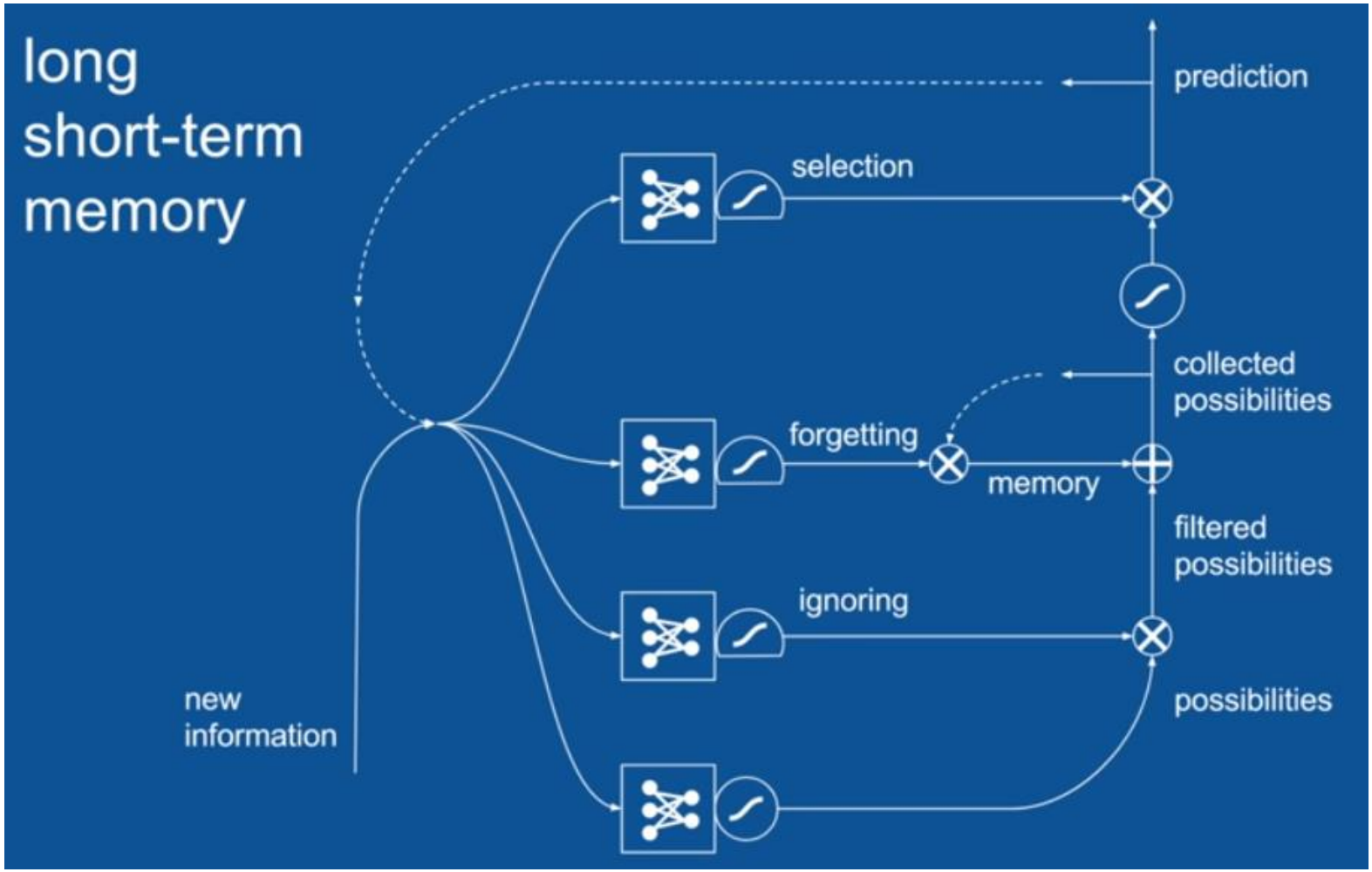

Plus junction: element-by-element addition
3
4
5
6
7
8
3 + 6
4 + 7
5 + 8
9
11
13

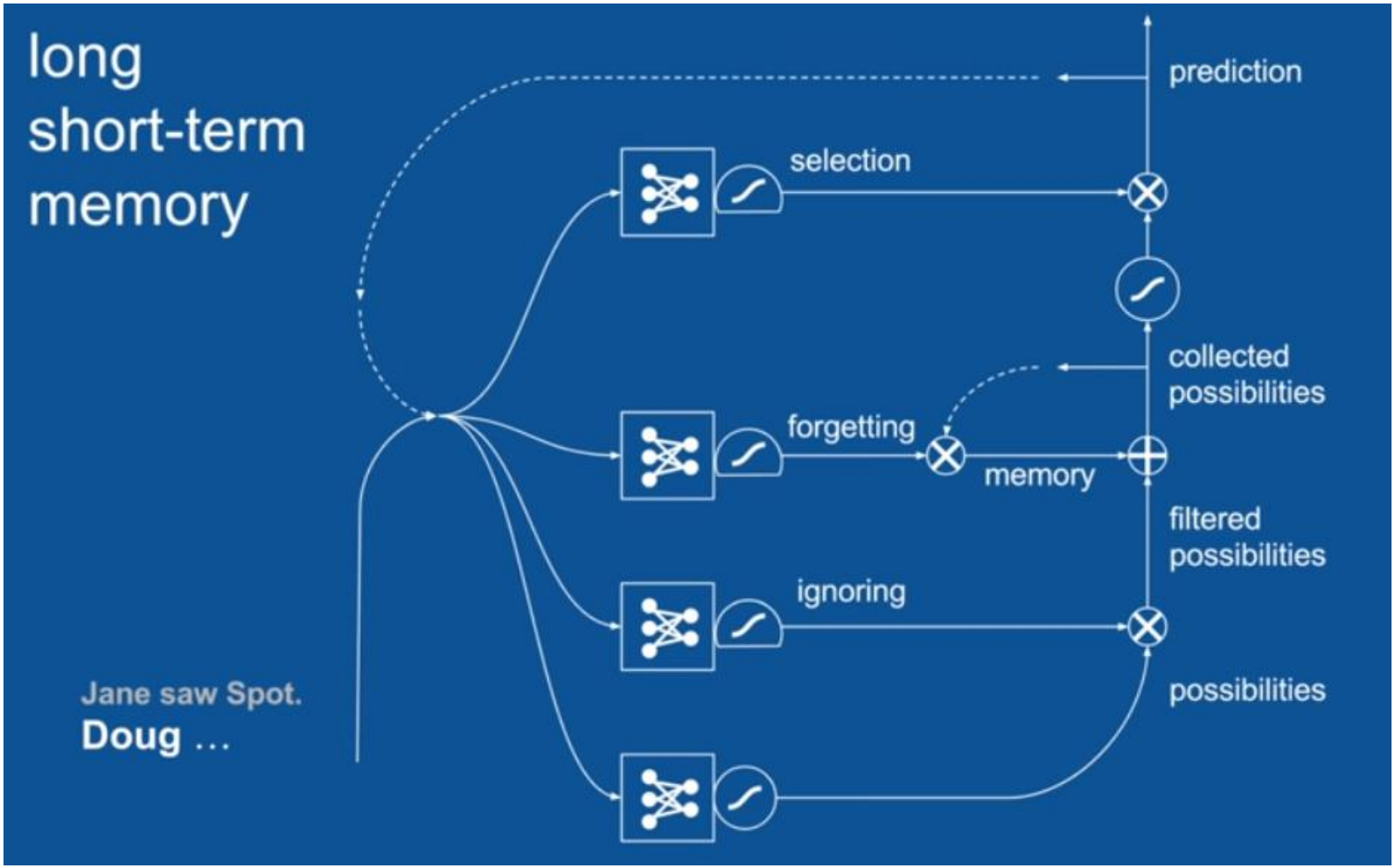

Times junction: element-by-element multiplication
3
4
5
6
7
8
3 x 6
4 x 7
5 x 8
18
28
40

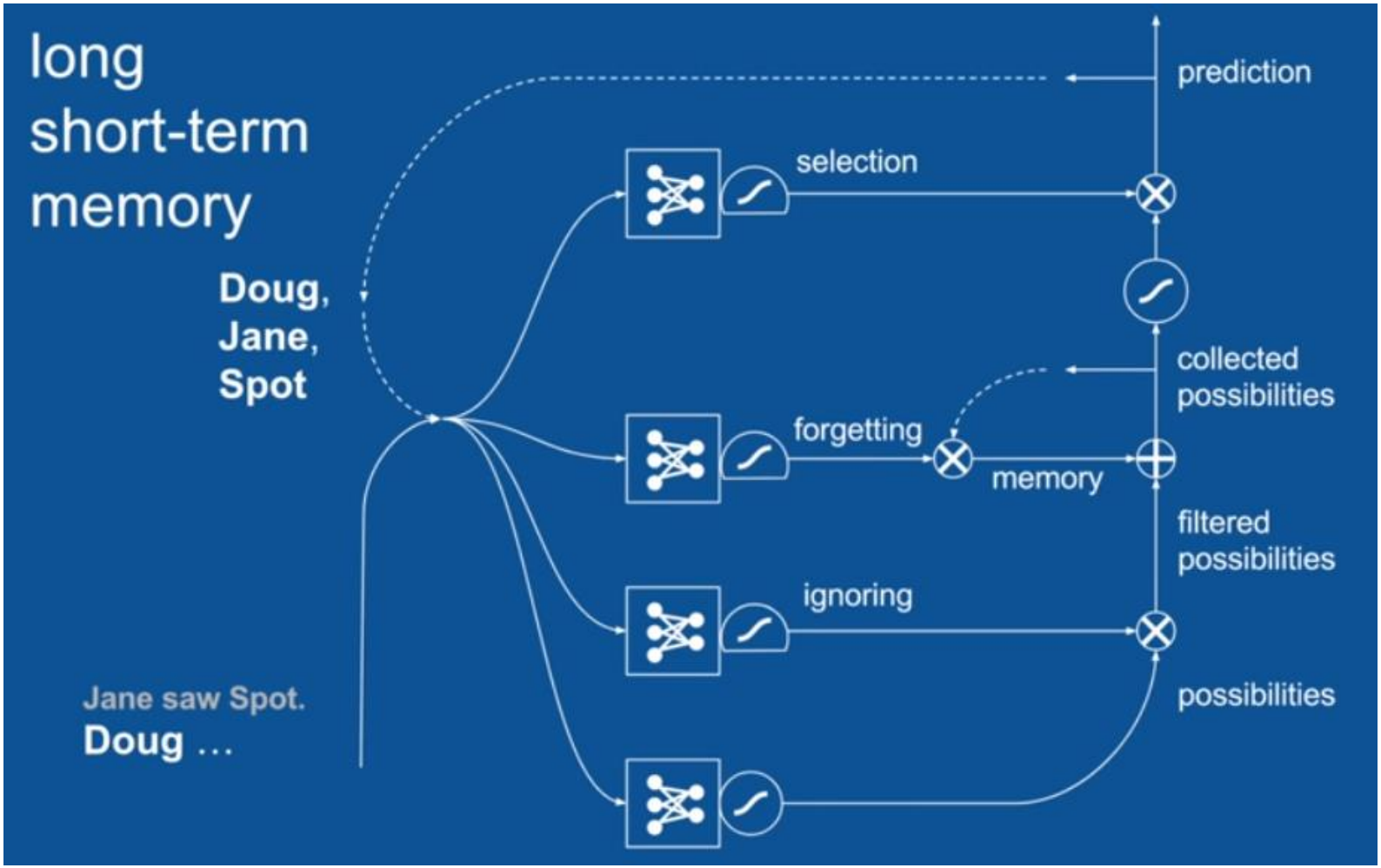

Gating
Signal
0.8
0.8
0.8
On / Off gating
1.0
0.5
0.0
0.8 x 1.0
0.8 x 0.5
0.8 x 0.0
0.8
0.4
0.0

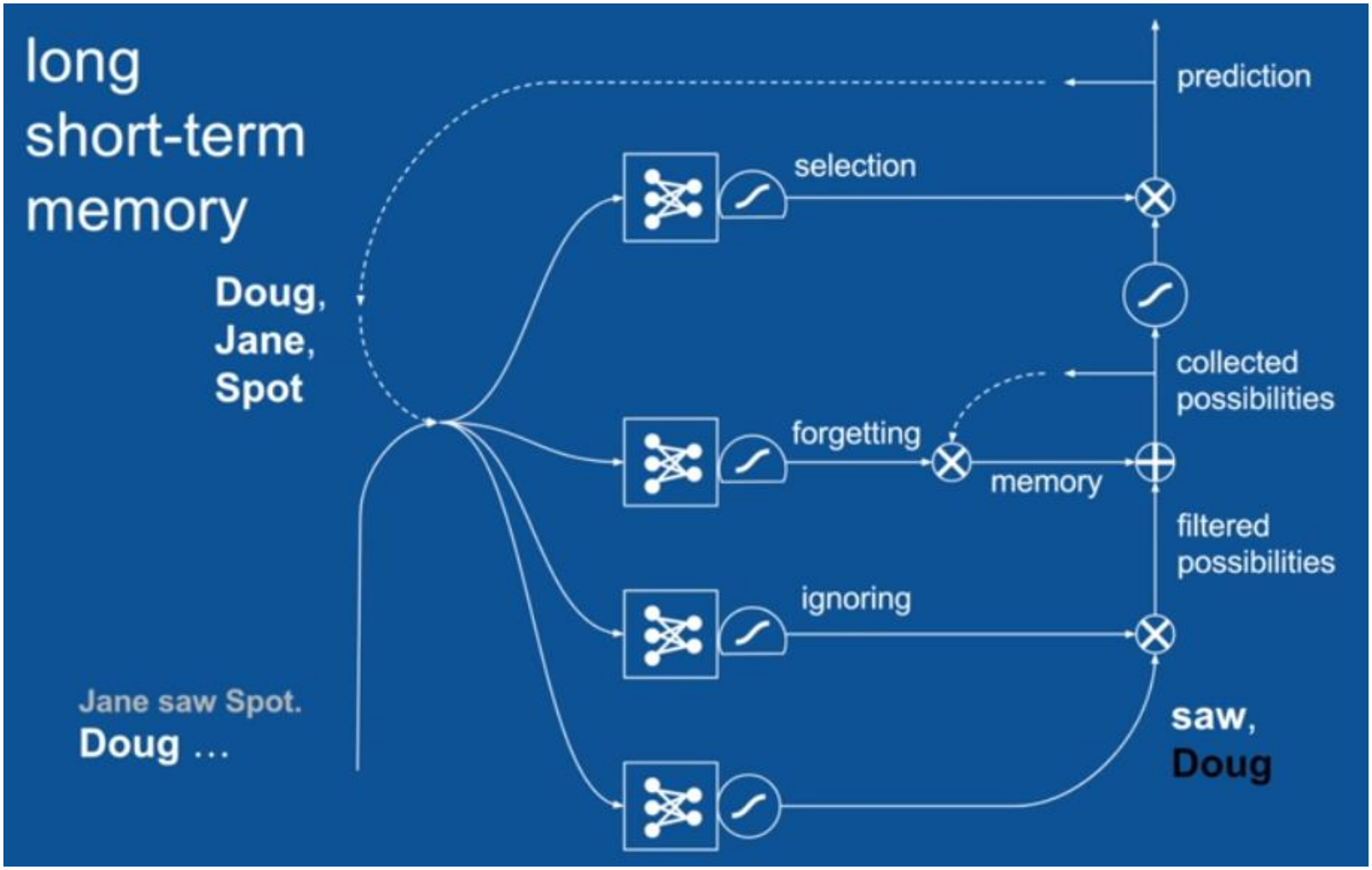

Logistic (sigmoid) squashing function
1.0
.5
-2.0
-1.0
1.0
2.0
No matter what you start with, the answer stays between 0 and 1.

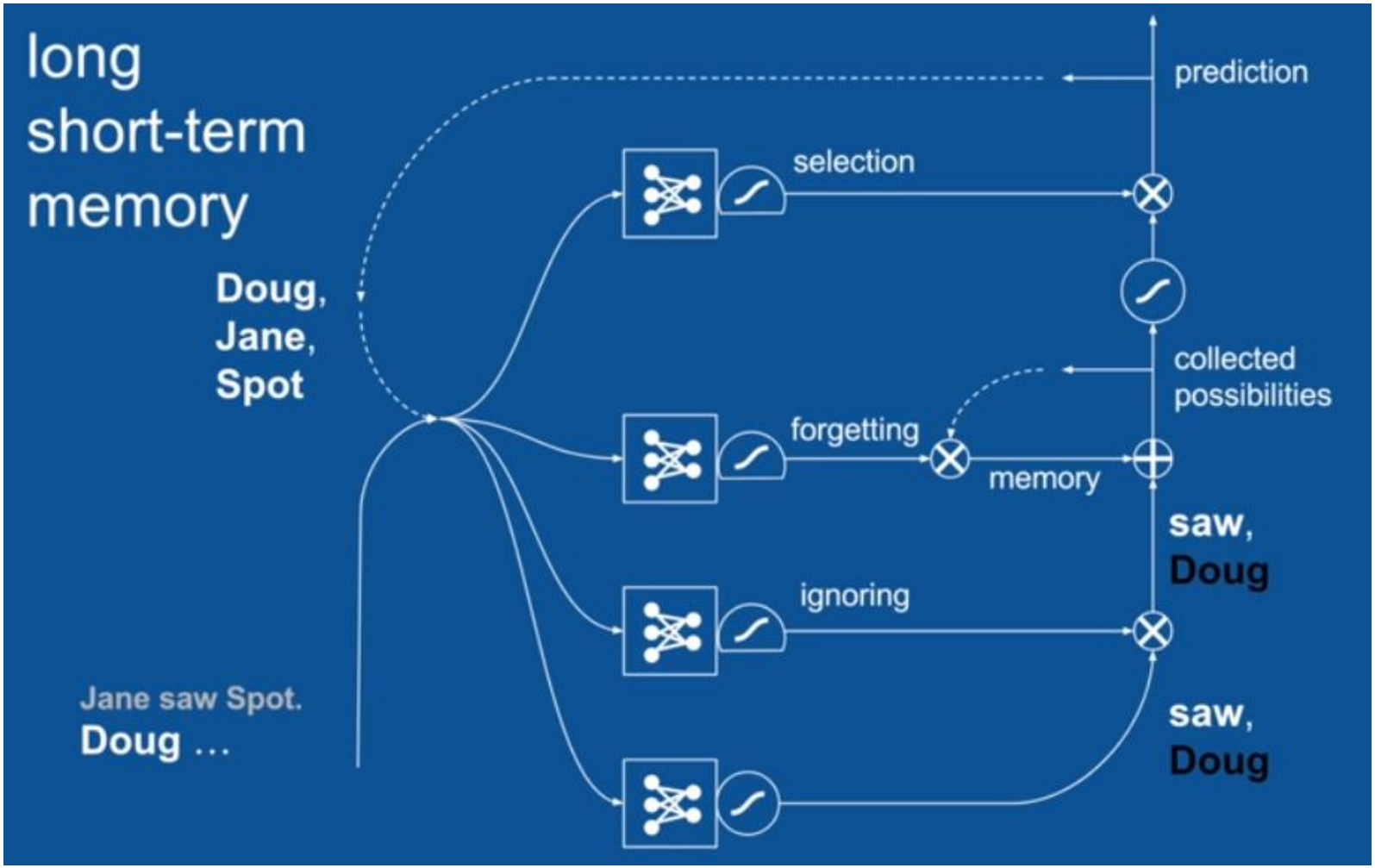

memory
prediction
selection
collected
possibilities
forgetting
memory
new
information
possibilities

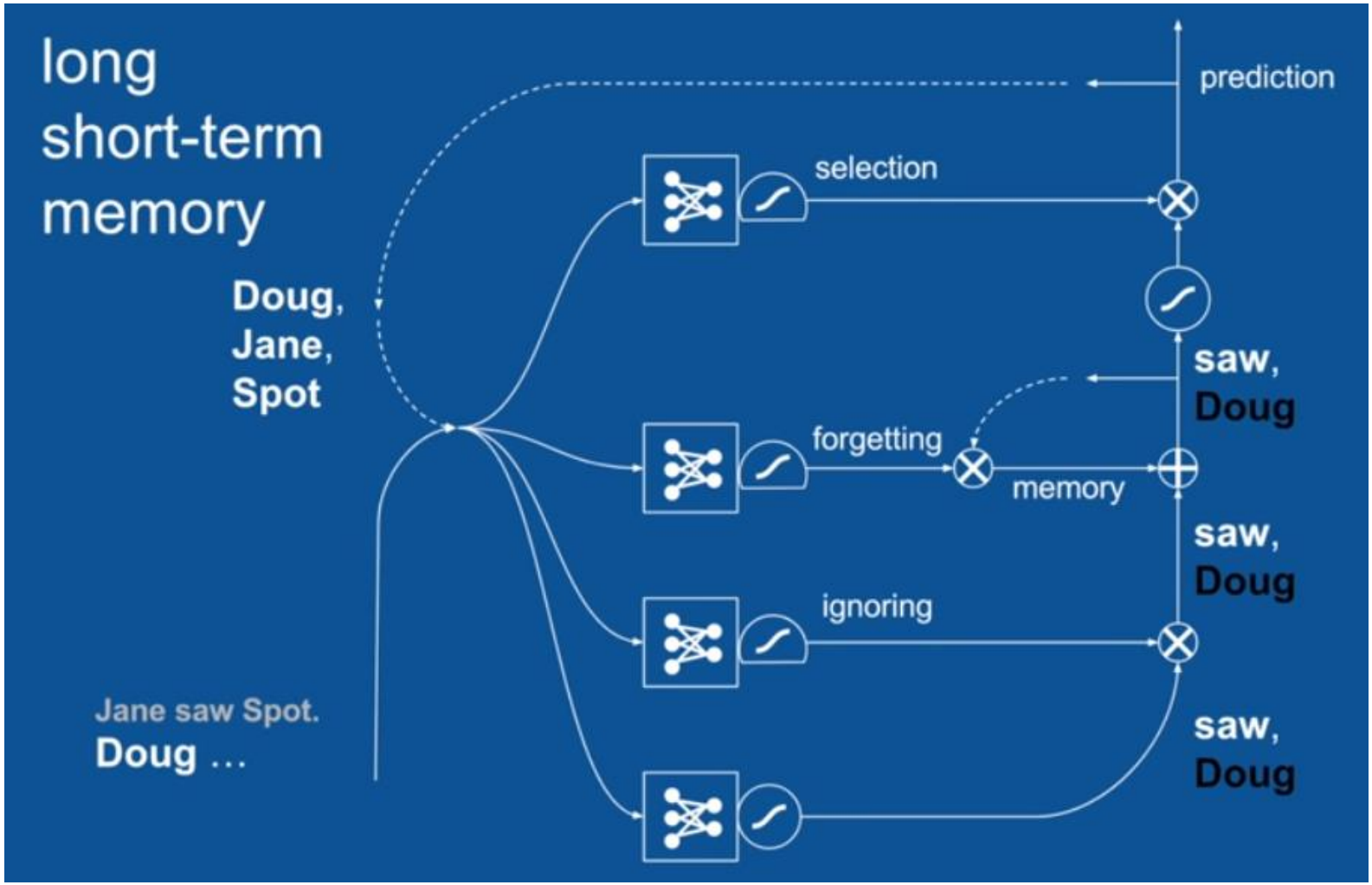

long
short-term
memory
prediction
selection
collected
possibilities
forgetting
memory
filtered
possibilities
ignoring
new
information
possibilities

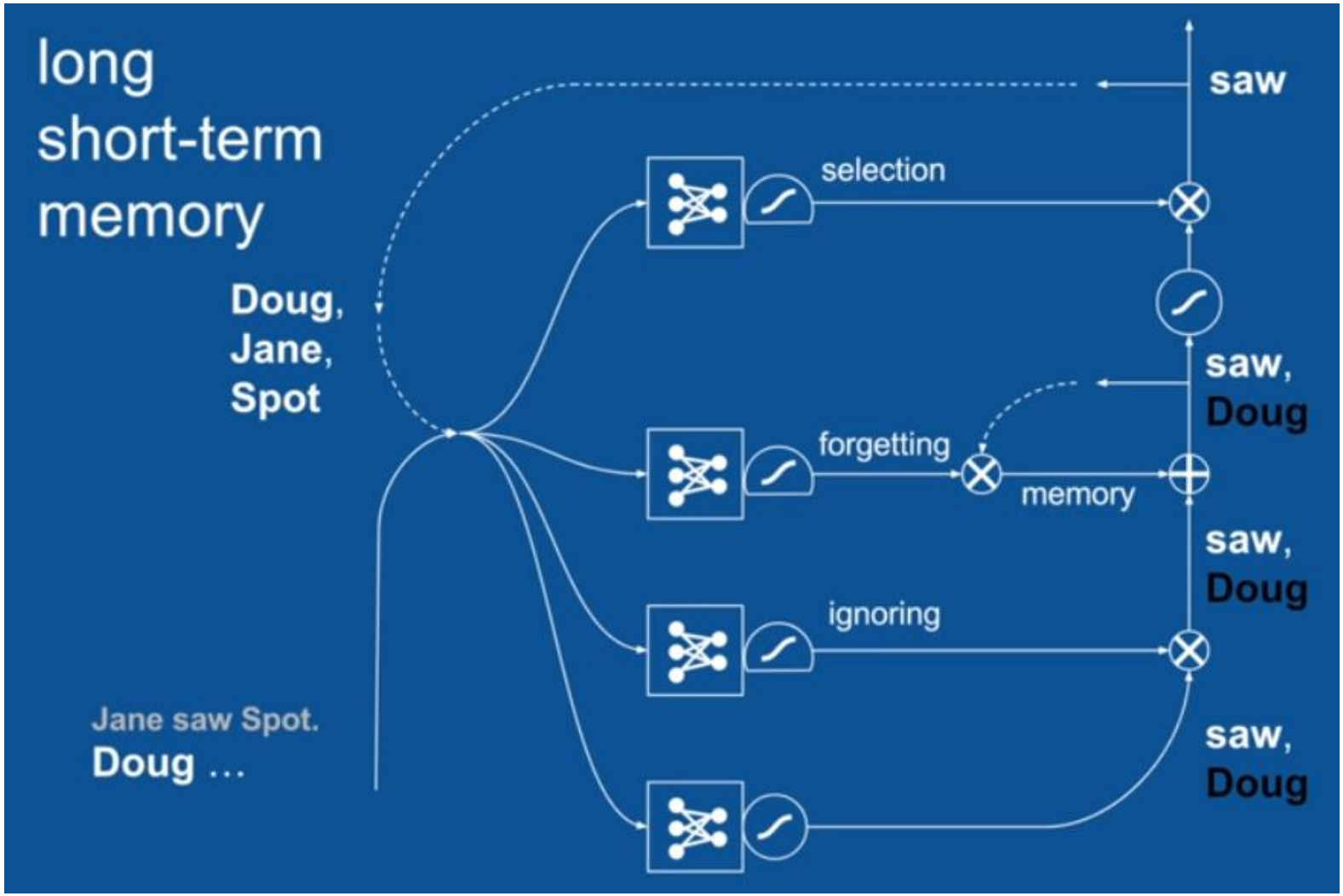

long
short-term
memory
prediction
selection
collected
possibilities
forgetting
memory
filtered
possibilities
ignoring
possibilities
Jane saw Spot.
Doug …

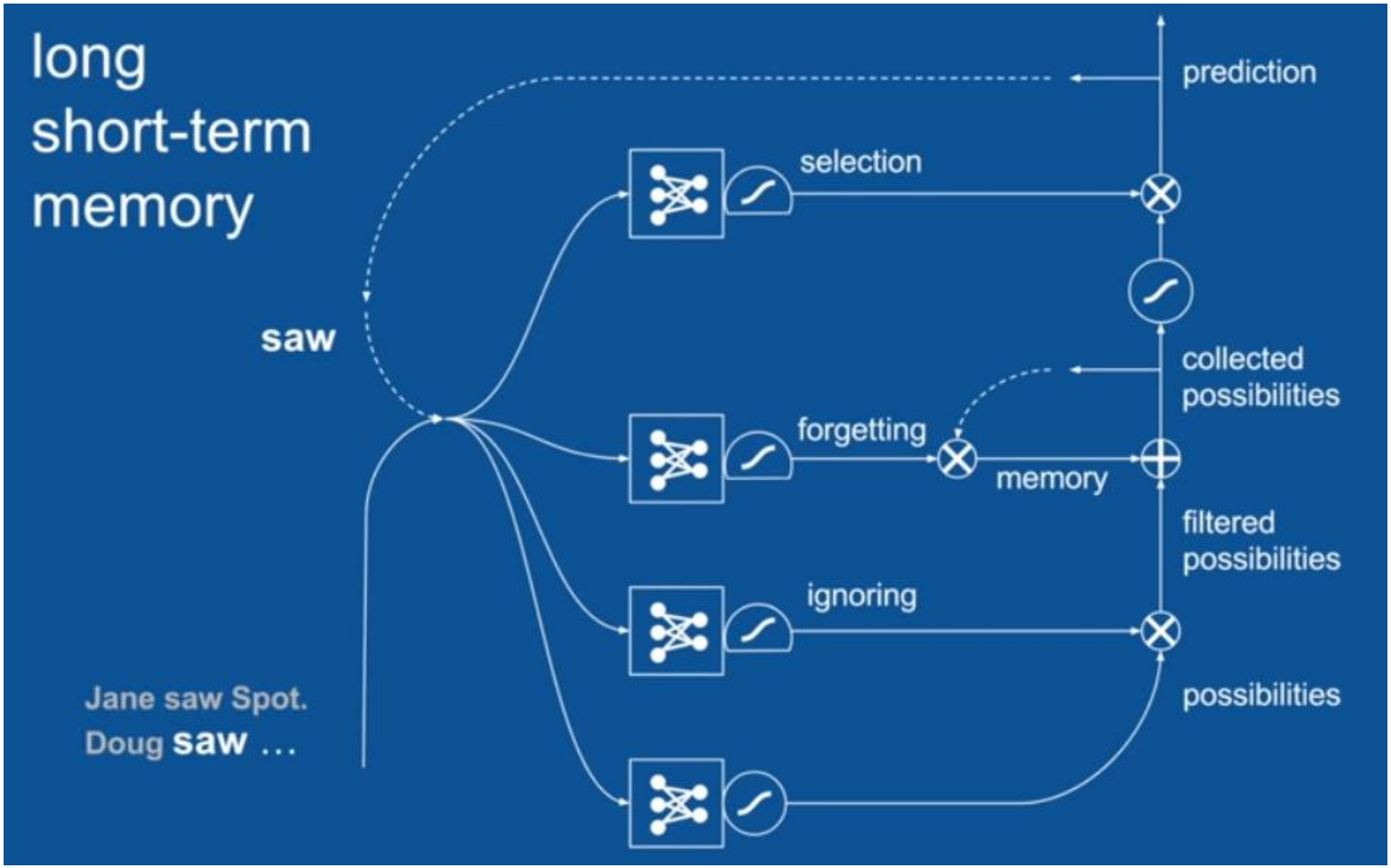

long
short-term
memory
Doug,
Jane,
Spot
Jane saw Spot.
Doug …
prediction
selection
collected
possibilities
forgetting
memory
filtered
possibilities
ignoring
possibilities

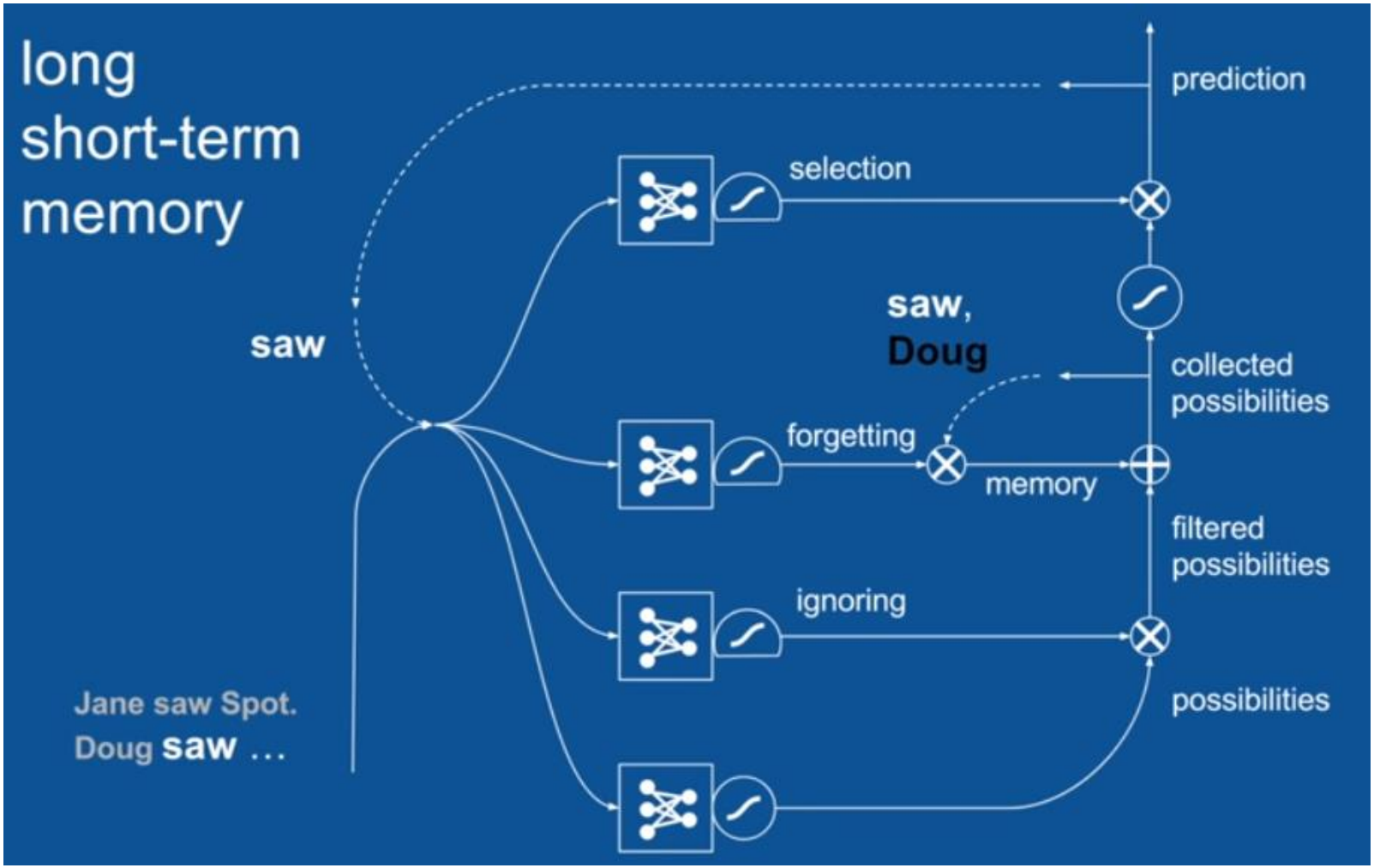

long
short-term
memory
Doug,
Jane,
Spot
prediction
selection
collected
possibilities
forgetting
memory
filtered
possibilities
ignoring
Jane saw Spot.
Doug …
saw,
Doug

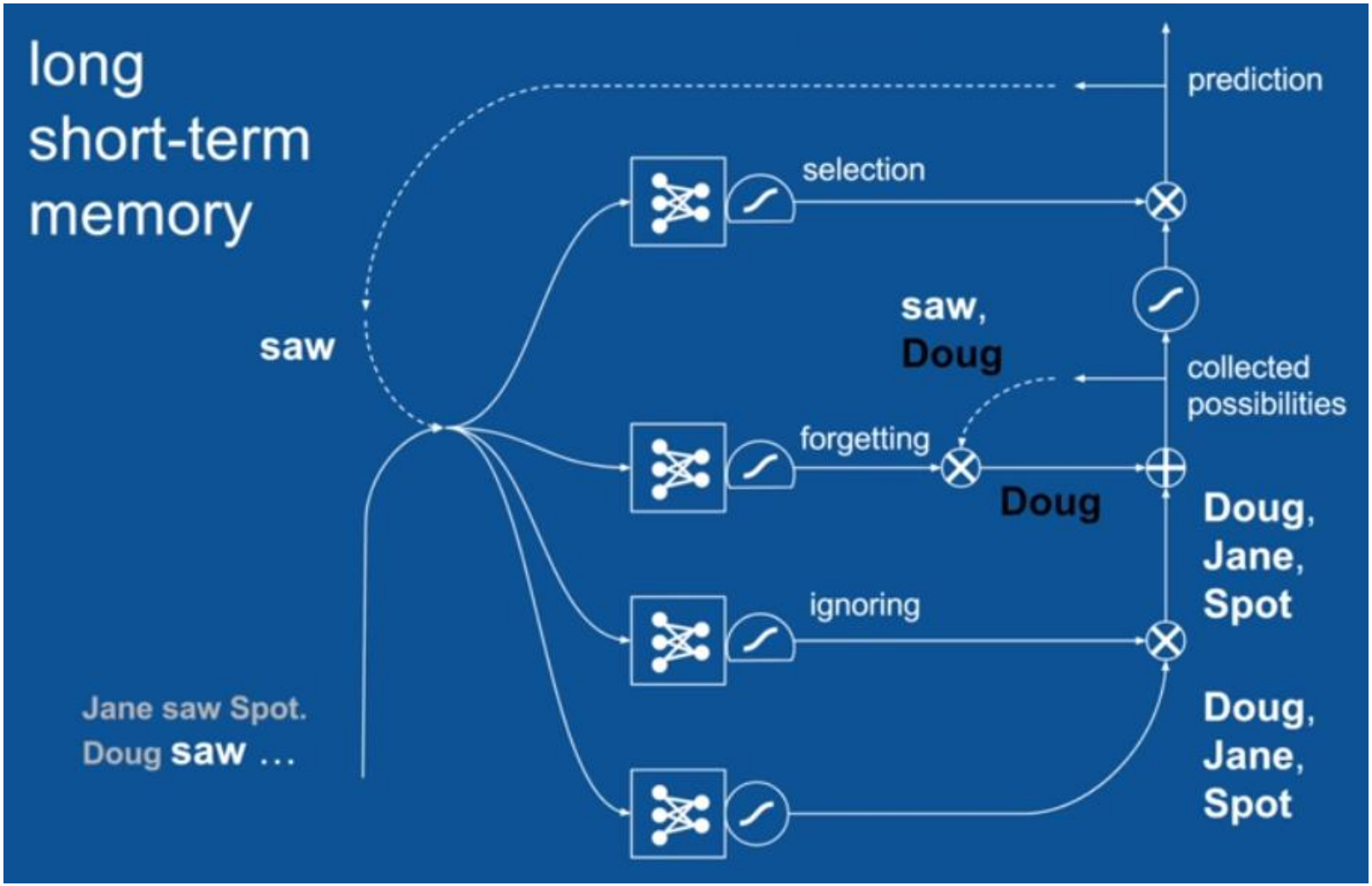

long
short-term
memory
Doug,
Jane,
Spot
prediction
selection
collected
possibilities
forgetting
memory
saw,
Doug
ignoring
saw,
Doug
Jane saw Spot.
Doug …

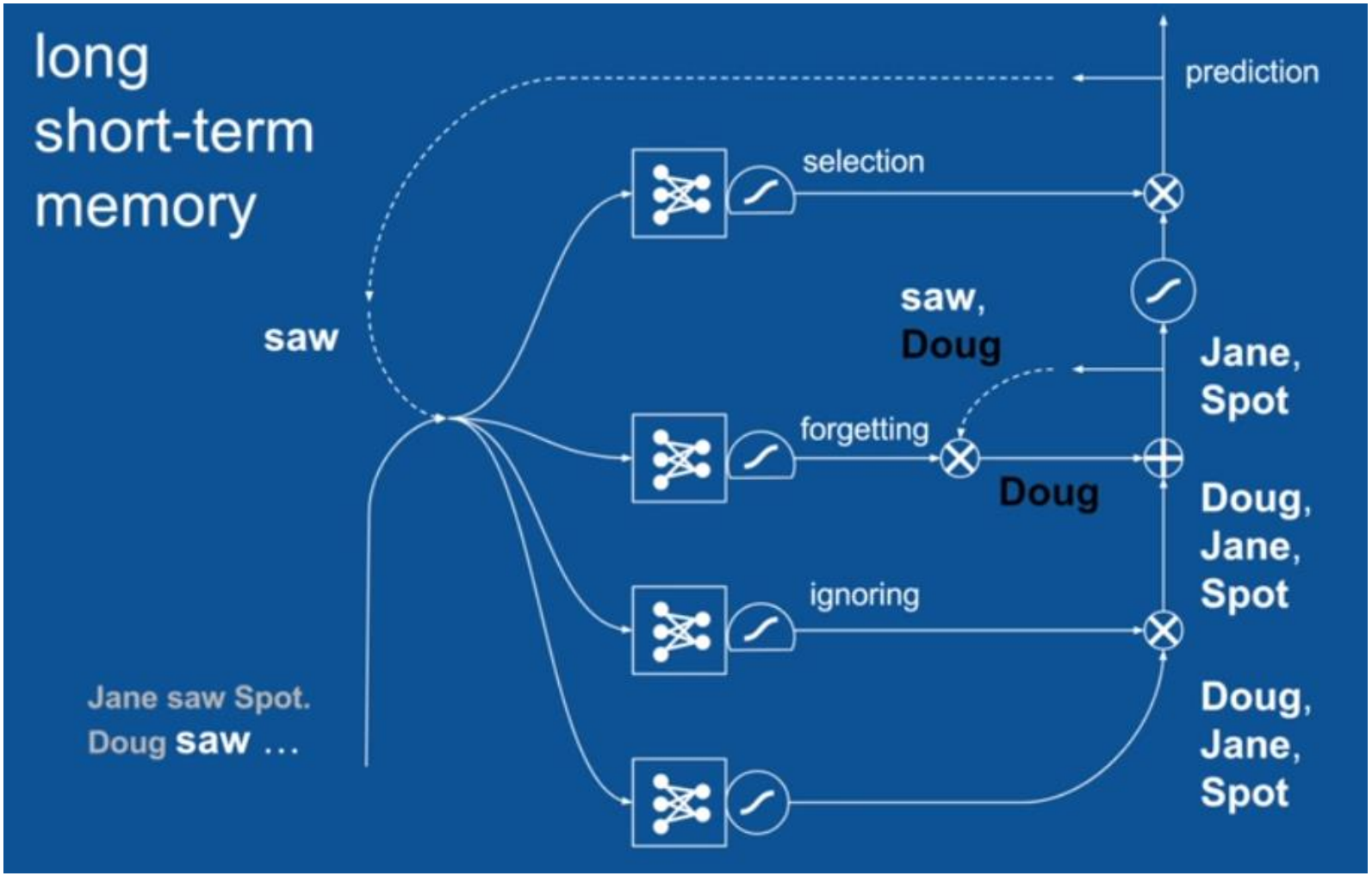

long
short-term
memory
Doug,
Jane,
Spot
prediction
selection
forgetting
memory
ignoring
saw,
Doug
saw,
Doug
saw,
Doug
Jane saw Spot.
Doug …

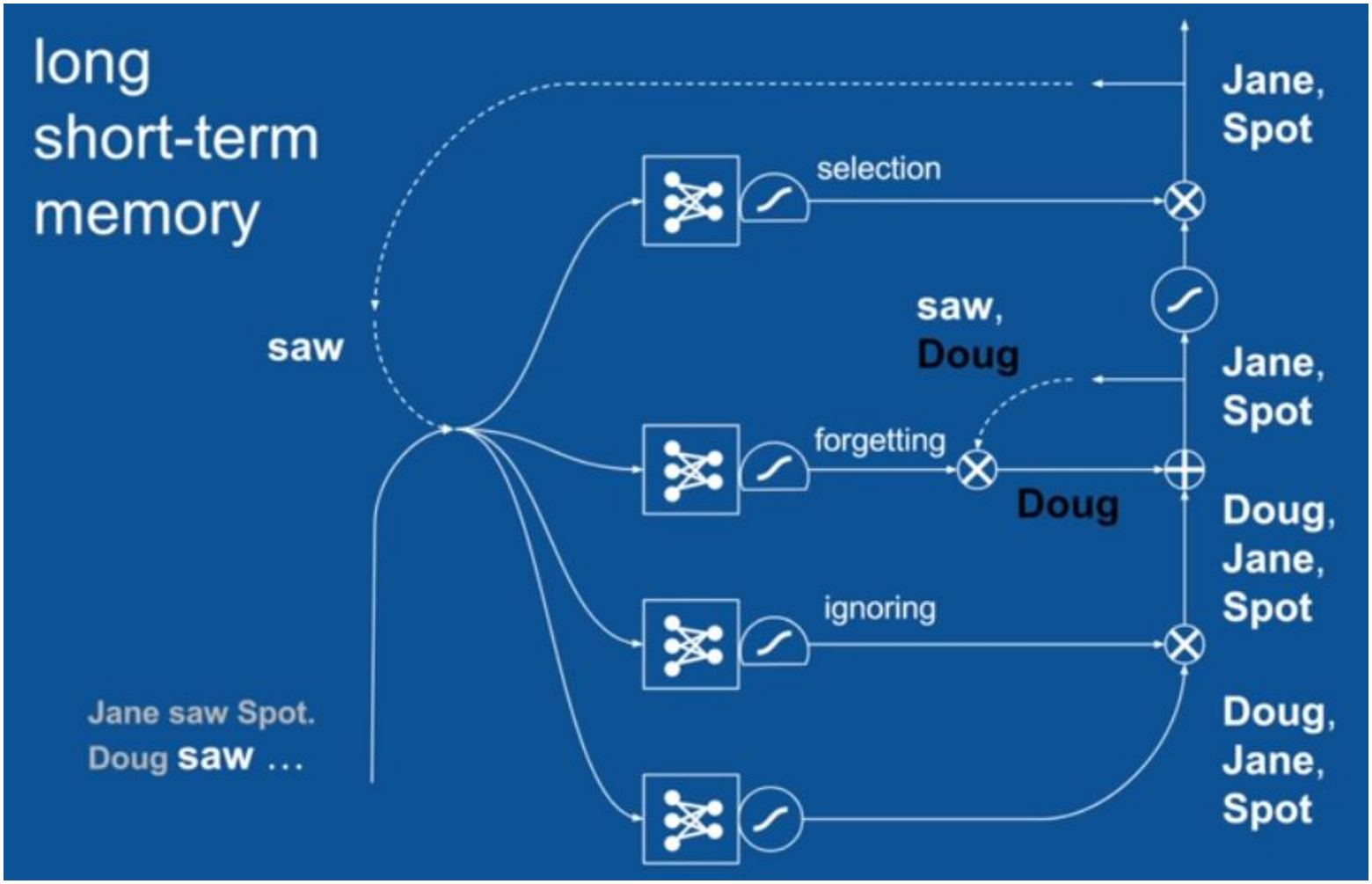

long
short-term
memory
Doug,
Jane,
Spot
selection
forgetting
memory
ignoring
saw
saw,
Doug
saw,
Doug
saw,
Doug
Jane saw Spot.
Doug …

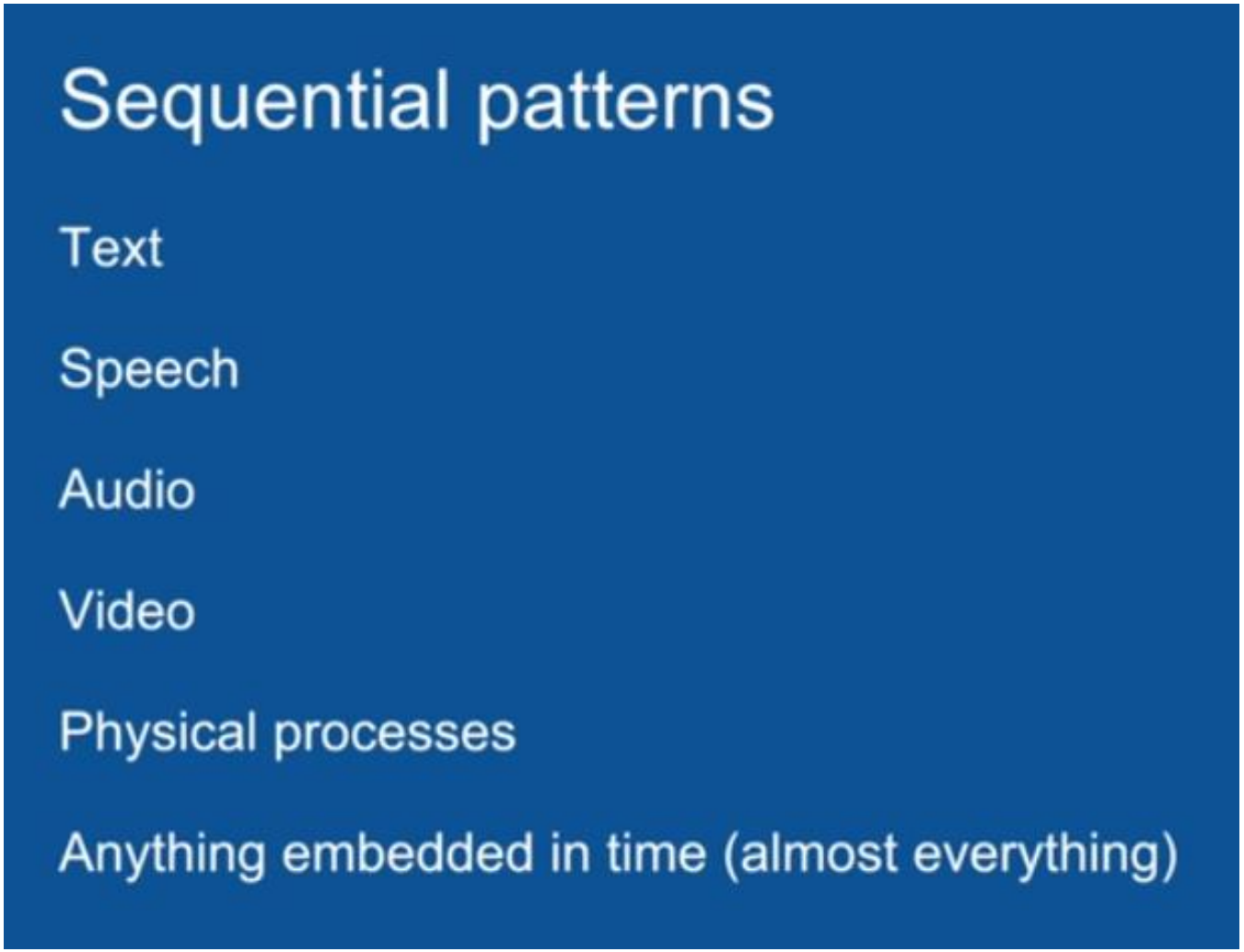

long
short-term
memory
saw
Jane saw Spot.
Doug saw …
selection
forgetting
ignoring
prediction
collected
possibilities
memory
filtered
possibilities
possibilities


long
short-term
memory
prediction
selection
saw,
Doug
saw
collected
possibilities
forgetting
memory
filtered
possibilities
ignoring
possibilities
Jane saw Spot.
Doug saw …


long
short-term
memory
prediction
selection
saw,
Doug
saw
collected
possibilities
forgetting
Doug
Doug,
Jane,
Spot
ignoring
Jane saw Spot.
Doug saw …
Doug,
Jane,
Spot


long
short-term
memory
prediction
selection
saw
saw,
Doug
Jane,
Spot
forgetting
Doug
Doug,
Jane,
Spot
ignoring
Jane saw Spot.
Doug saw …
Doug,
Jane,
Spot


long
short-term
memory
saw
selection
forgetting
ignoring
saw,
Doug
Doug
Jane,
Spot
Jane,
Spot
Doug,
Jane,
Spot
Doug,
Jane,
Spot
Jane saw Spot.
Doug saw …


Sequential patterns
Text
Speech
Audio
Video
Physical processes
Anything embedded in time (almost everything)

4 Compreendendo o bloco LSTM

- Grande parte do material desta seção foi extraído do blog de Christopher Olah (http://colah.github.io/posts/2015-08-Understanding-LSTMs/).
- Redes neurais recorrentes apresentam realimentação de informação, o que significa que uma certa informação pode persistir ao longo do tempo.
- Para o caso específico de um bloco de neurônios que apresenta realimentação externa, é possível representar este bloco como uma sequência de múltiplas cópias de si mesmo, cada cópia fornecendo uma entrada ao bloco sucessor na sequência.

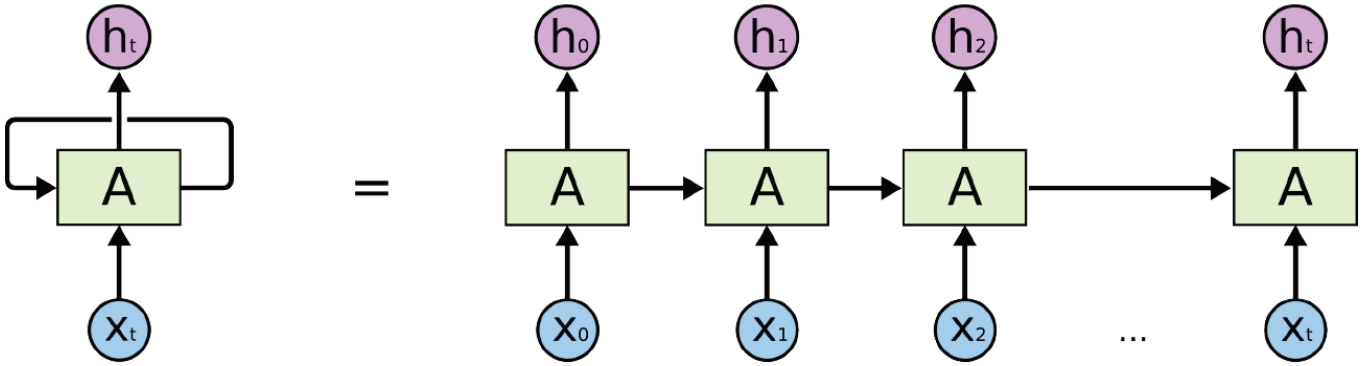


Figura 5 – Rede neural recorrente e o seu desdobramento no tempo

- Fica evidente, portanto, que redes neurais recorrentes estão intimamente vinculadas a aplicações que envolvem contexto e informação sequencial. De fato, há um elenco consistente e expressivo de aplicações bem-sucedidas de redes neurais recorrentes em reconhecimento de fala, modelagem de linguagem, tradução de textos e rotulação de imagens.
- No entanto, até a proposição das LSTMs (*Long Short Term Memories*) em 1997 (HOCHREITER & SCHMIDHUBER, 1997), o ajuste de parâmetros em redes recorrentes convencionais se mostrava bem mais desafiador que no caso não-recorrente. Além disso, a capacidade de implementar simultaneamente memórias de curto, médio e longo prazos, necessária em várias aplicações, era bem limitada. Portanto, os blocos LSTMs são responsáveis por grande parte do sucesso recente das redes neurais recorrentes.
- Redes formadas por um ou mais blocos LSTMs podem aprender dependências de prazos arbitrários, particularmente de longo prazo, sem muito esforço.

- Uma rede neural recorrente convencional apresenta um bloco na forma ilustrada na Figura 6, exibida na versão com desdobramento no tempo e tal que o módulo em amarelo realiza um mapeamento multidimensional não-linear.

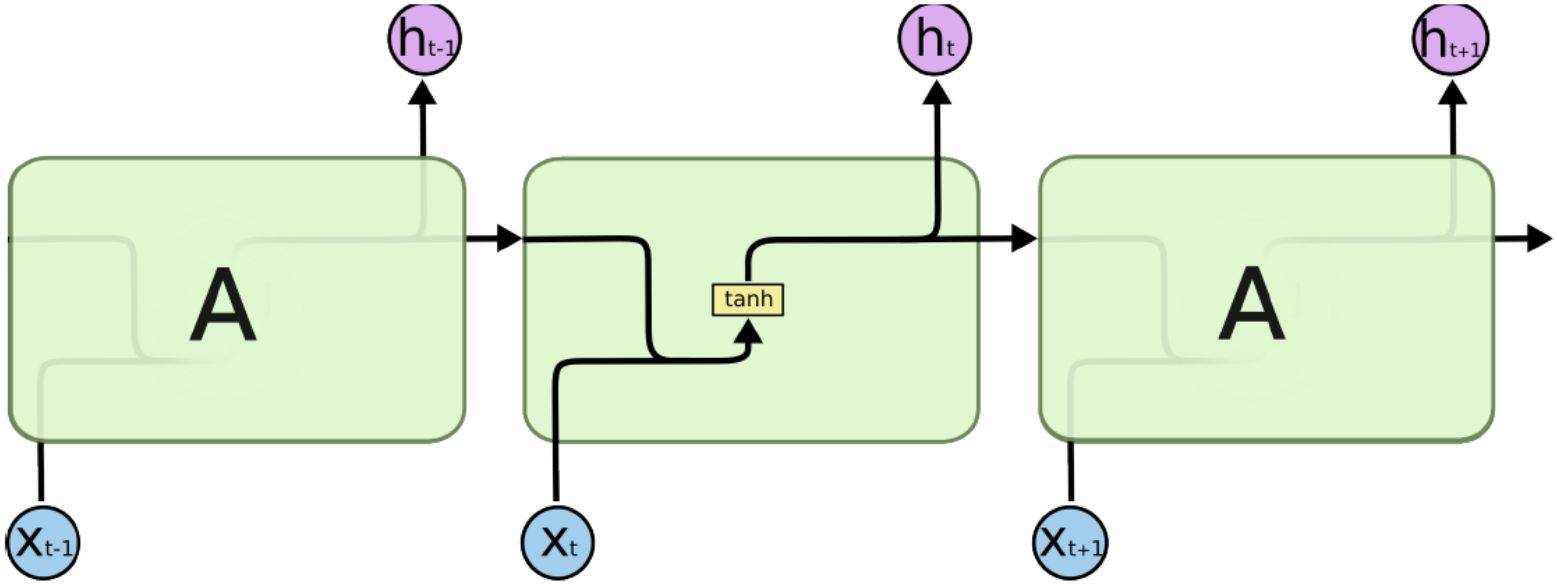


Figura 6 – Bloco de uma rede neural recorrente convencional, ou seja, com realimentação externa.

- Repare que o bloco é bastante inflexível, pois só é capaz de preservar informações de curto prazo, mais especificamente, o sinal produzido no instante anterior.

- Portanto, para memórias de longo prazo, sua implementação ficava vinculada à existência de um número equivalente de blocos, em sequência (repare que não se está falando aqui do desdobramento no tempo e sim blocos recorrentes consecutivos). Logo, a duração da memória ficava associada à arquitetura. Mais ainda, supondo a necessidade de implementação conjunta de memórias de prazos diferentes, todos esses prazos deveriam ser contemplados a priori na arquitetura da rede neural, não sendo, portanto, passíveis de aprendizado de acordo com as demandas da aplicação.
- Já o bloco LSTM (ver Figura 7) possui uma estrutura mais sofisticada, composta por quatro mapeamentos multidimensionais não-lineares, sendo três deles portas (*gates*) que permitem implementar memórias de prazo arbitrário (e com um prazo específico para cada dimensão do vetor de sinais) em um único bloco LSTM.

- De acordo com a notação apresentada, o fluxo de sinais é vetorial e as operações são realizadas elemento a elemento do vetor de sinais. Além disso, os retângulos amarelos são mapeamentos multidimensionais não-lineares.

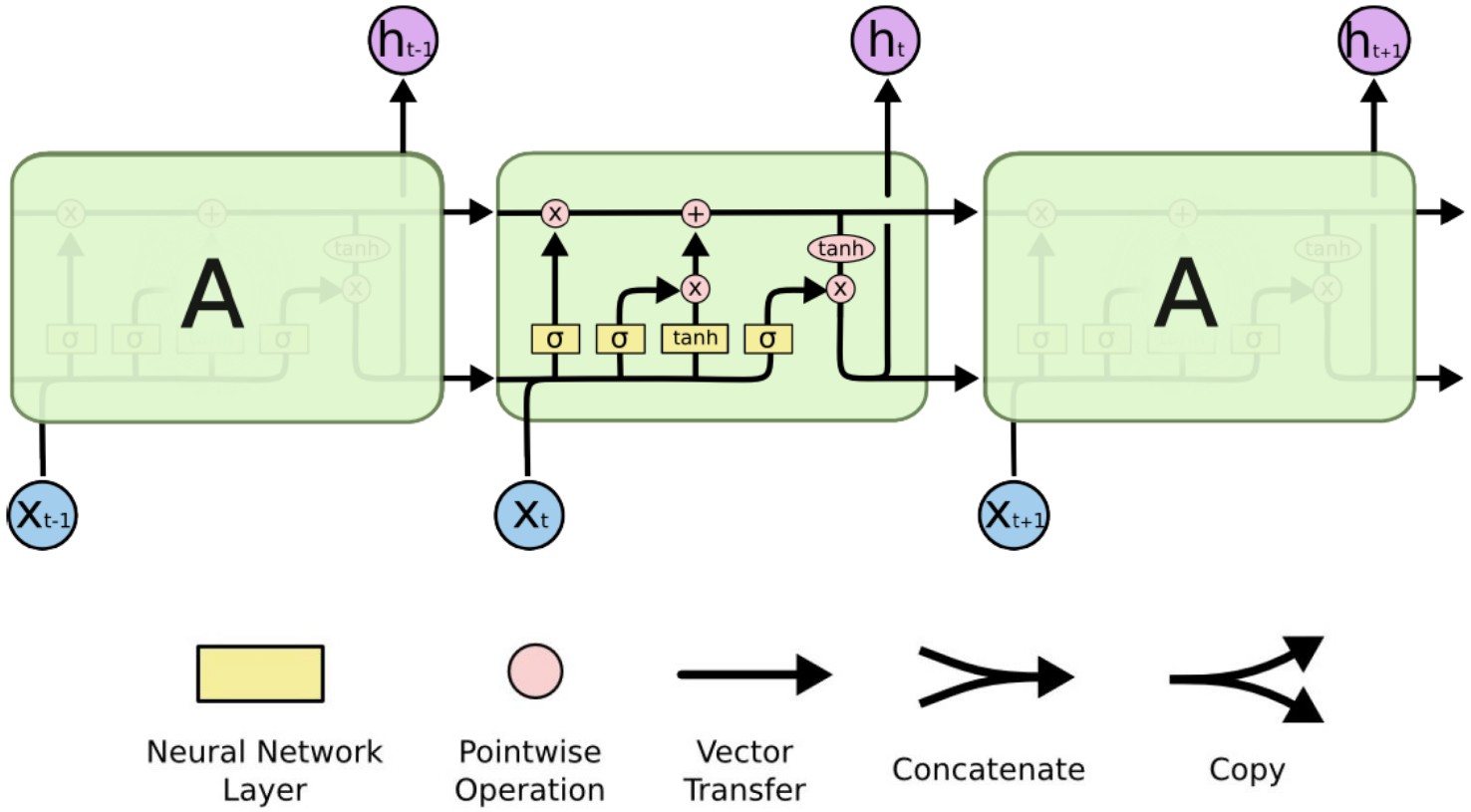


Figura 7 – Bloco LSTM e sua notação associada

- Como em toda estrutura recorrente, o aspecto-chave do bloco LSTM é o seu vetor de estados, que corresponde ao sinal presente na linha horizontal em destaque na Figura 8.

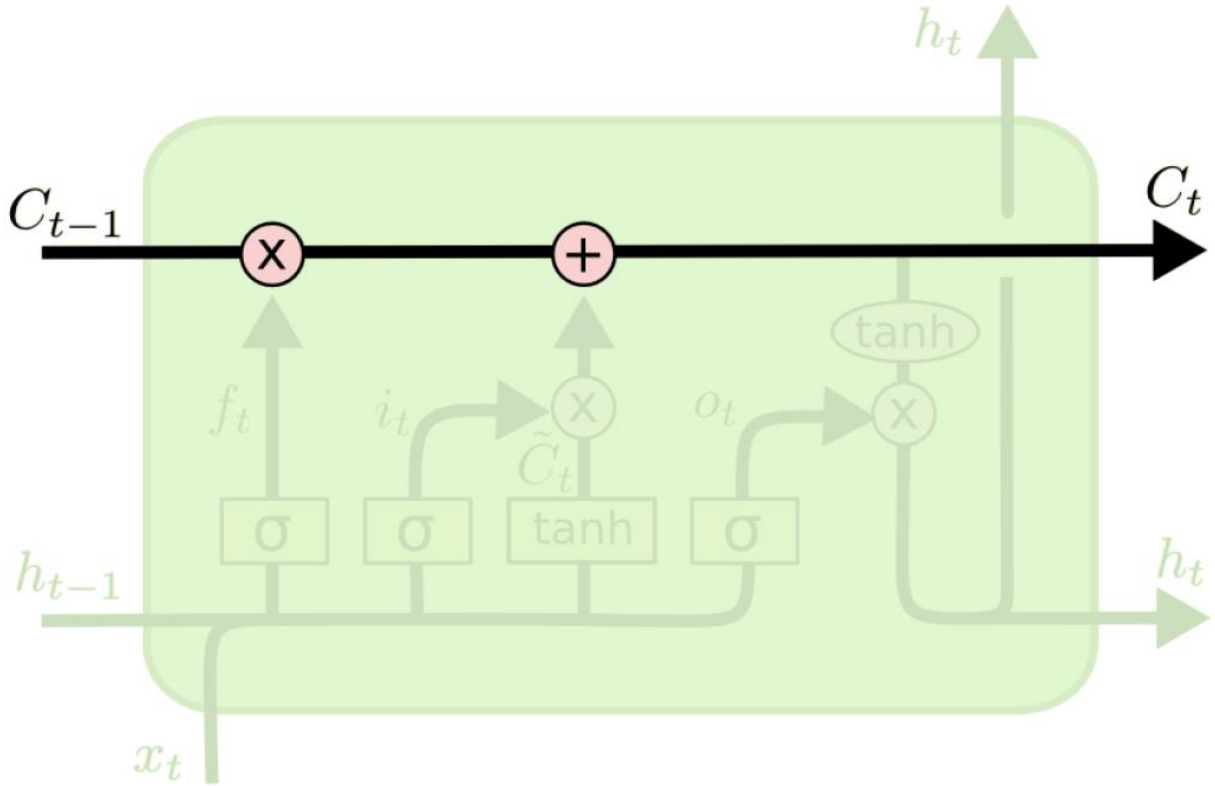
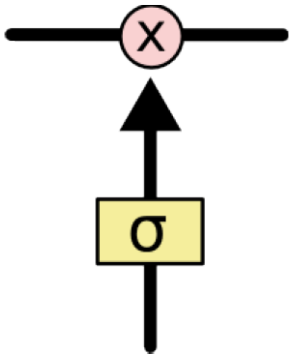


Figura 8 – Destaque para o vetor de estados da LSTM

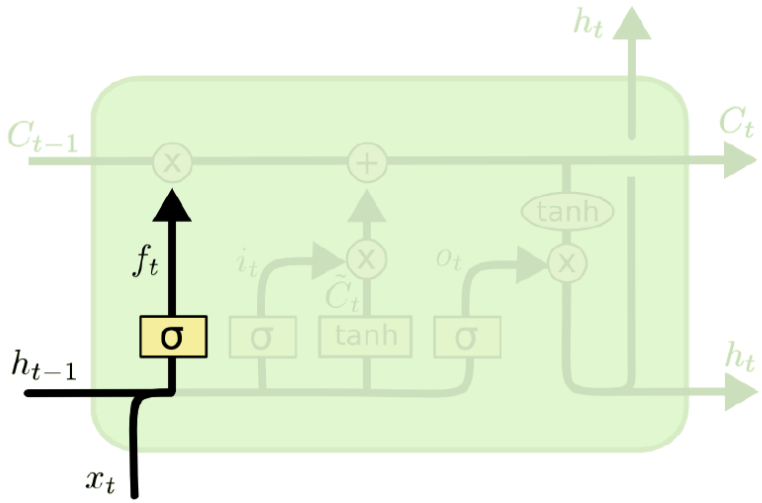
- O vetor de estados pode passar por três operações algébricas simples, permitindo remover informação do vetor de estados ou adicionar informação ao vetor de estados. Logo, caso nenhuma remoção ou adição ocorra, o vetor de estados do bloco LSTM se mantém inalterado de um instante de tempo ao seguinte.
- Essas operações sobre o vetor de estados do bloco LSTM são comandadas por três portas cujo comportamento é definido durante o processo de treinamento.



- Cada elemento do vetor de saída de uma porta pode assumir um valor no intervalo [0,1]. Valores iguais a 0 removem toda a informação daquele canal, enquanto que valores iguais a 1 mantêm toda a informação naquele canal.

5 O passo-a-passo do bloco LSTM

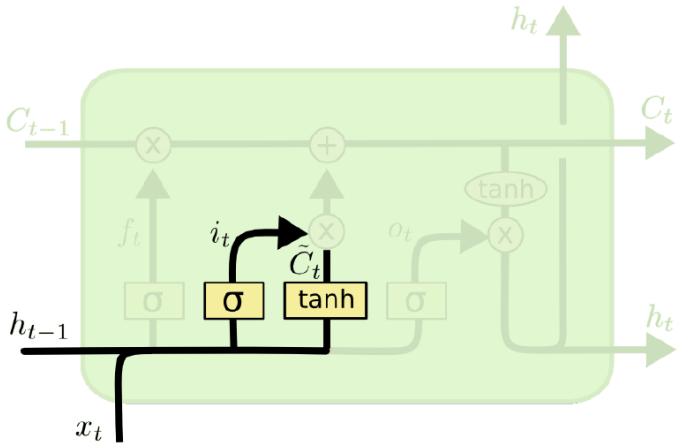
- O papel da primeira porta é indicar o que deve ser preservado para o instante *t* do vetor de estados C_{t-1} do bloco LSTM, elemento a elemento.
- Dadas as entradas h_{t-1} e x_t, elas são mapeadas no vetor de saídas f_t, onde cada um de seus elementos corresponde a um valor no intervalo [0,1]. Esta porta é denominada de porta do esquecimento (*forgetting gate*).



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figura 9 – Destaque para a porta do esquecimento

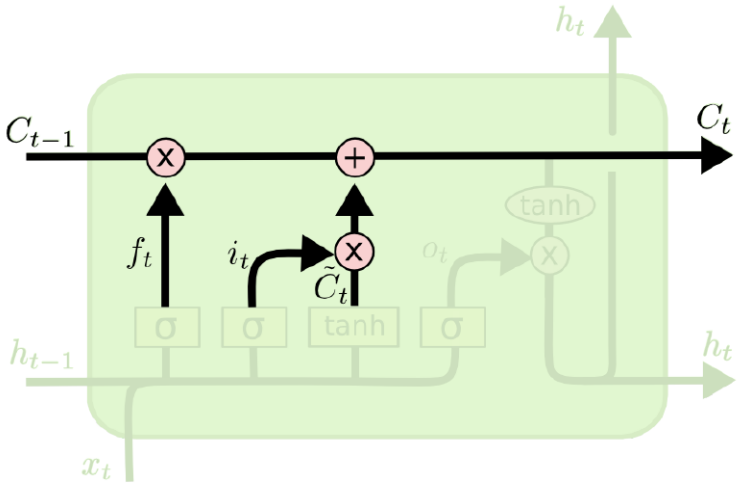
- O próximo passo é decidir que informação nova será adicionada ao vetor de estados C_{t-1} para se produzir C_t. Isso é feito em duas partes: (1) Uma porta decide em que grau (valor no intervalo [0,1]) os elementos do vetor de estado serão atualizados, sendo denominada porta de entrada (foi chamada de *ignoring gate* na seção anterior); (2) Uma camada de neurônios com função de ativação tangente hiperbólica fornece a informação nova $\tilde{C}_t$ (valor no intervalo [–1,+1]) a ser adicionada ao vetor de estados, de acordo com a indicação da porta de entrada.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figura 10 – Destaque para a porta de entrada e para a nova informação

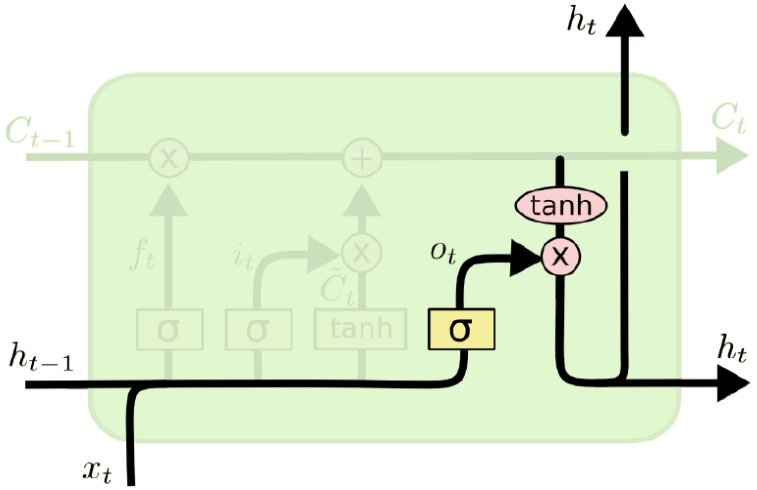
- O novo vetor de estados C_t do bloco LSTM é definido pela **composição aditiva** das operações das Figuras 9 e 10, conforme indicado na Figura 11.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figura 11 – Definição do novo vetor de estados

- A saída do bloco LSTM não vai ser diretamente o seu novo vetor de estados. Primeiro, o vetor de estados C_t é mapeado para o intervalo [–1,+1]. Em seguida, a porta de saída (foi chamada de *selection gate* na seção anterior) vai indicar com que grau (valor no intervalo [0,1]) cada elementos de C_t será liberado.



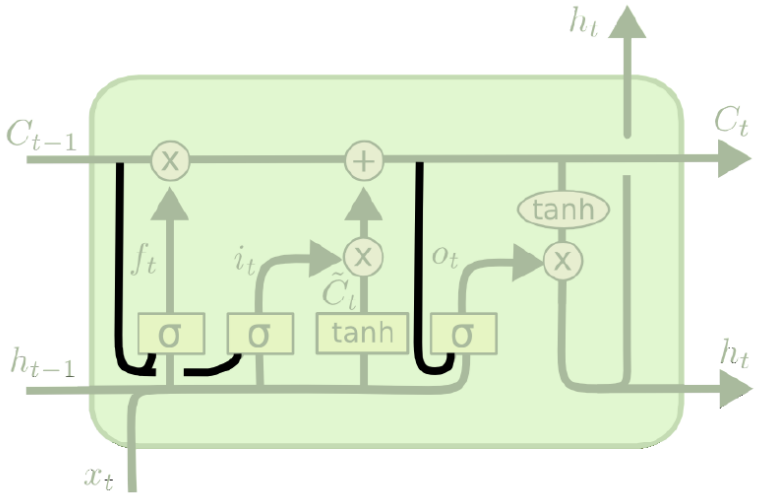
$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Figura 12 – Destaque para a porta de saída e para o sinal de saída h_t efetivamente liberado pelo bloco LSTM

6 Variantes do bloco LSTM

- Existem propostas na literatura que diferem da versão convencional apresentada até aqui, ao inserir fluxos adicionais de informação. O propósito é trazer ganhos em aplicações específicas, mas à custa de um maior desafio computacional na fase de treinamento.

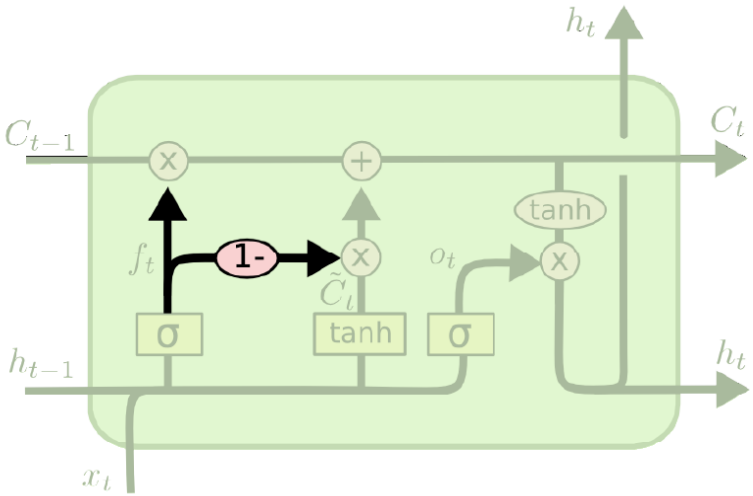


$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Figura 13 – Bloco LSTM que considera o vetor de estados como uma entrada adicional para as três portas

- Uma variação bastante utilizada é aquela que acopla os graus de esquecimento e de inovação, ou seja, acopla os papéis das portas de esquecimento e de entrada, de modo que elas operem de modo complementar. Com isso, cada elemento do vetor de estados vai estar sujeito a uma operação complementar entre esquecimento e inovação.

- Neste caso, só há esquecimento caso haja informação nova a ser inserida em seu lugar. Em outras palavras, só se adiciona conteúdo novo caso haja conteúdo antigo a ser esquecido, na mesma proporção.



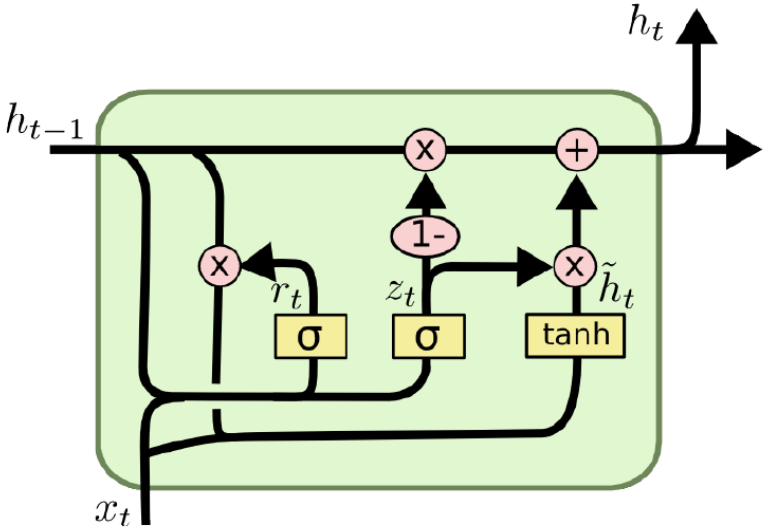
$$C_t = f_t * C_{t-1} + (1 - \boldsymbol{f_t}) * \tilde{C}_t$$

Figura 14 – Bloco LSTM com operação complementar entre as portas de esquecimento e de entrada

- Uma variação mais acentuada do bloco LSTM é denominada GRU (*Gated Recurrent Unit*) e foi introduzida em 2014 (CHO et al., 2014). Seu treinamento é

mais simples e esta proposta vem se tornando bastante popular, devido aos bons resultados comparativos.

- A GRU combina as portas de esquecimento e de entrada numa porta de atualização e também faz com que o vetor de estados passe a ser o vetor de saídas. Outras modificações também estão presentes.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figura 15 – Bloco GRU (*Gated Recurrent Unit*)

7 Referências bibliográficas

CHO, K., VAN MERRIENBOER, B., GULCEHRE, C., BAHDANAU, D., BOUGARES, F., SCHWENK, H., BENGIO, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, arXiv:1406.1078, 2014.

GREFF, K., SRIVASTAVA, R.K., KOUTNÍK, J., STEUNEBRINK, B.R., SCHMIDHUBER, J. LSTM: A Search Space Odyssey, *IEEE Trans. on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222-2232, 2017.

HAYKIN, S. Neural Networks and Learning Machines, 3rd edition, Prentice Hall, 2008.

HOCHREITER, S., SCHMIDHUBER, J. Long short-term memory, Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

MCDONNELL, J.R., WAGEN, D. Evolving Recurrent Perceptrons for Time-Series Modeling. *IEEE Transactions Neural Networks*, vol. 5, no. 1, pp. 24-38, 1994.

NERRAND, O., ROUSSEL-GAGOT, P., URBANI, D., PERSONNAZ, L., DREYFUS, G. Training recurrent neural networks: Why and how? An illustration in dynamical process modeling. *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 178-184, 1994.

PEARLMUTTER, B.A. Gradient calculations for dynamic recurrent neural networks: a survey. *IEEE Transactions on Neural Networks*, vol. 6, no. 5, pp. 1212-1228, 1995.

PINEDA, F.J. Generalization of back-propagation to recurrent neural networks. *Physical Review Letters*, vol. 59, no. 19, pp. 2229-2232, 1987.

PINEDA, F.J. Recurrent Backpropagation and the Dynamical Approach to Adaptive Neural Computation. *Neural Computation*, vol. 1, no. 2, pp. 161-172, 1989.

SJÖBERG, J., ZHANG, Q., LJUNG, L., BENVENISTE, A., DELYON, B., GLORENNEC, P., HJALMARSSON, H., JUDITSKY, A. Nonlinear Black-box Modelling in System Identification: a Unified Overview, *Automatica*, vol. 31, no. 12, pp. 1691-1724, 1995.

WERBOS, P.J. Backpropagation Through Time: What It Does and How to Do It. *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550-1560, 1990

WILLIAMS, R. J., ZIPSER, D. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks, *Neural Computation*, vol. 1, no. 2, pp. 270-280, 1989.