Redes Neurais Recorrentes

1. Introdução

Embora as redes neurais *feedforward* sejam poderosas ferramentas de transformação de representação, capazes de resolver tarefas bastante complexas em reconhecimento de padrões e visão computacional, elas constroem mapeamentos estáticos da entrada para a saída. Isso significa que, diante de um padrão de entrada $\mathbf{x}(t)$, a rede não reutiliza a informação processada dos padrões anteriores $\mathbf{x}(t-1), \mathbf{x}(t-2), ...$, gerando sempre a mesma saída, independente do "momento" (ou contexto) em que a entrada aparece.

Por outro lado, redes neurais recorrentes (RNNs, do inglês *recurrent neural networks*) contêm ciclos que propagam ativações de neurônios em instantes passados como

entradas para influenciar as previsões no instante atual. Estas ativações ficam armazenadas nos estados internos da rede, os quais, em tese, podem guardar informações de contexto temporal de longo prazo. Este mecanismo permite que RNNs explorem uma janela de contexto que varia dinamicamente ao longo do histórico da sequência de entrada.

Considere, inicialmente, um neurônio cuja saída seja realimentada em sua entrada, como mostra a Figura 1(a). Para cada padrão $\mathbf{x}(t)$, a resposta gerada pelo neurônio pode ser genericamente representada como:

$$y(t) = \varphi(\mathbf{x}(t), y(t-1); \boldsymbol{\theta}), \tag{1}$$

em que $\boldsymbol{\theta}$ denota o conjunto de parâmetros do neurônio. Note que, devido à presença de $y(t-1)$, tem-se uma relação de recorrência, de modo que, implicitamente, $y(t)$ é função do histórico de entradas $\mathbf{x}(t-1), \mathbf{x}(t-2), \ldots$. É

possível representar esta pequena rede ao longo do tempo, conforme ilustra a Figura 1(b).

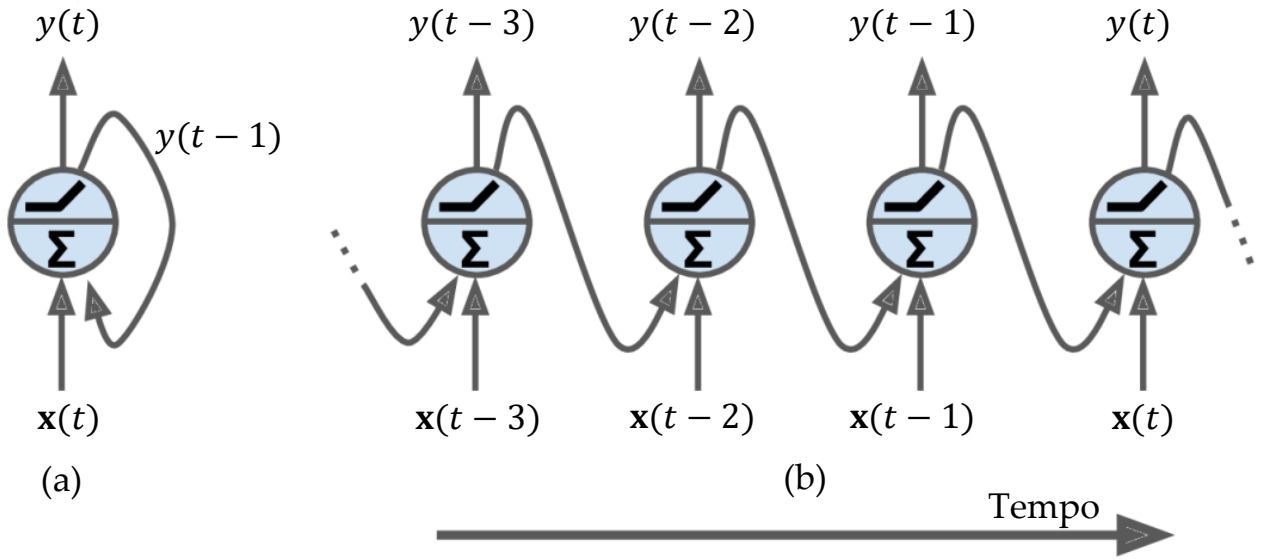

Figura 1 – Exemplo de um neurônio com recorrência. Adaptada de (GÉRON, 2019).

Note que há uma cópia do neurônio para cada instante de tempo, sendo que as conexões recorrentes levam a saída do neurônio para as cópias seguintes, associadas ao próximo instante de tempo. Este procedimento é denominado desdobramento

(*unfold* ou, também, *unrolling*) da rede, e tem um papel importante no treinamento de RNNs.

Uma camada recorrente com este tipo de realimentação é mostrada na Figura 2, juntamente o seu desdobramento no tempo.

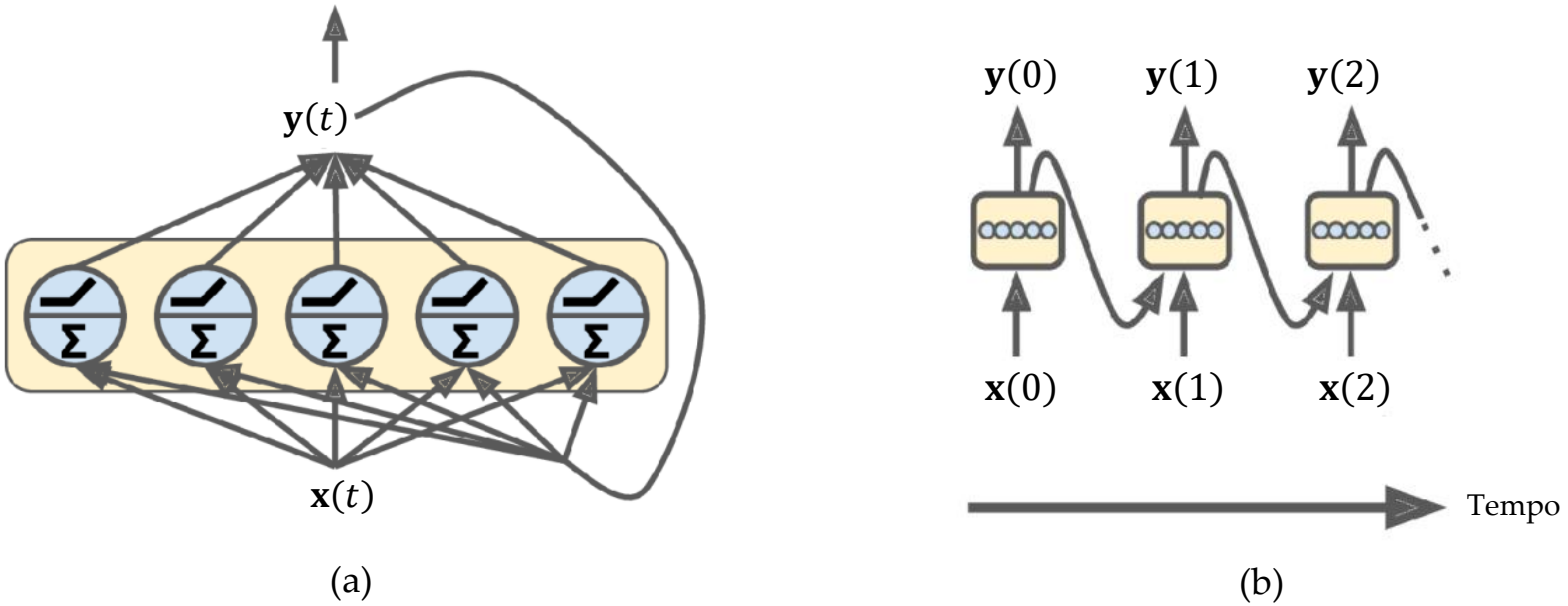

Figura 2 – Rede neural recorrente e seu desdobramento no tempo. Adaptada de (GÉRON, 2019).

Dada uma instância de entrada $\mathbf{x}(t)$, o vetor de saída produzido pela camada recorrente, considerando neurônios do tipo *perceptron*, pode ser escrita como:

$$\mathbf{y}(t) = \varphi\left(\mathbf{W}_x\mathbf{x}(t) + \mathbf{W}_y\mathbf{y}(t-1) + \mathbf{b}\right), \quad (2)$$

onde as matrizes $\mathbf{W}_x$ e $\mathbf{W}_y$ especificam os pesos das conexões de entrada e das realimentações, respectivamente.

Uma vez que a saída em um instante t é, implicitamente, um reflexo de todas as entradas em instantes passados, a rede possui, de fato, uma memória. A parte da rede neural que preserva um estado ao longo do tempo, o qual condensa toda a informação do histórico de eventos, é chamada de *célula* (ou *bloco*) *de memória*. A rede descrita em (2) representa um tipo básico de célula, capaz de aprender apenas padrões temporais de curto prazo. Mais adiante, veremos uma célula mais flexível com maior poder de memorização. Em termos gerais, o estado de uma célula de memória, denotado por $\mathbf{h}(t)$, é função das entradas atuais $\mathbf{x}(t)$ e do estado anterior $\mathbf{h}(t-1)$. Ademais, a saída também é função do estado anterior e das entradas atuais:

$$\mathbf{h}(t) = f\big(\mathbf{x}(t), \mathbf{h}(t-1)\big)$$
$$\mathbf{y}(t) = g\big(\mathbf{x}(t), \mathbf{h}(t-1)\big) \tag{3}$$

A Figura 3 exibe uma célula genérica de memória, bem como o desdobramento no tempo desta estrutura.

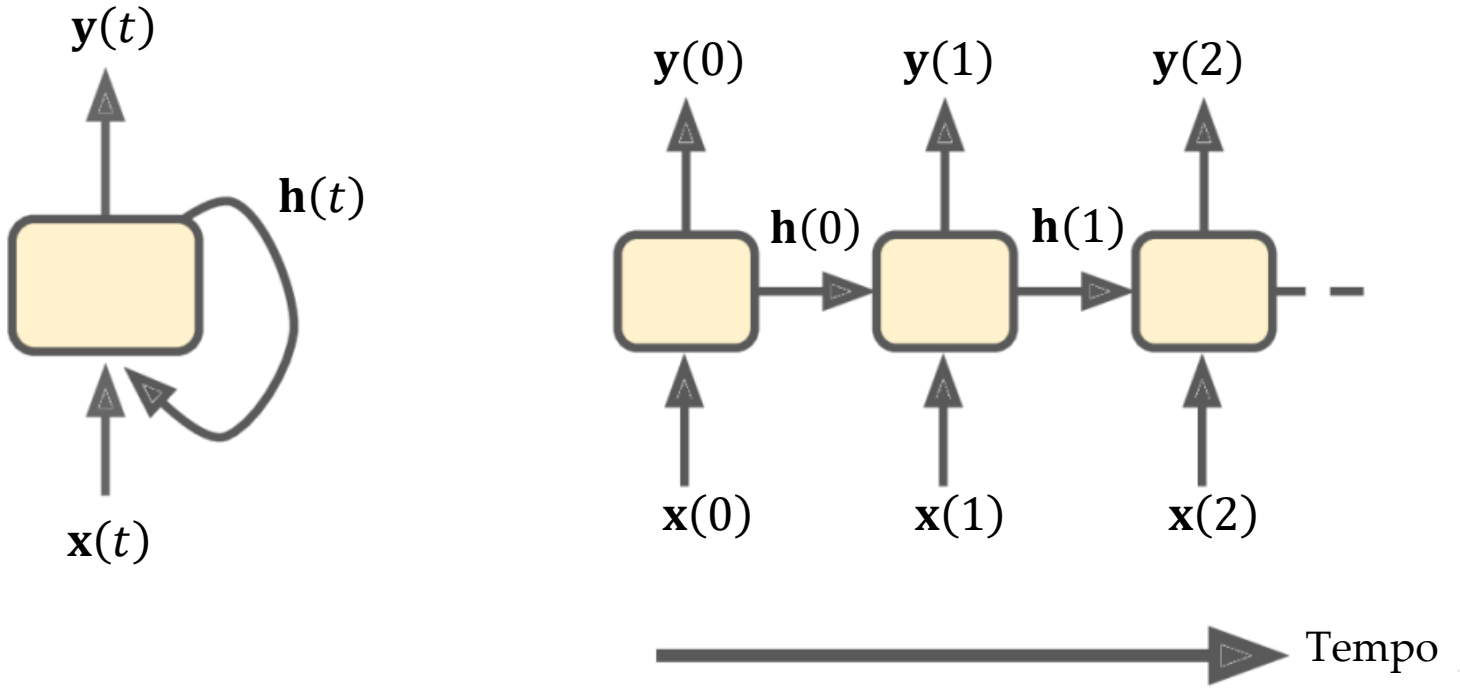

Figura 3 – Célula de memória, com destaque para o vetor de estado e o desdobramento no tempo. Adaptada de (GÉRON, 2019).

1.1. Processo de treinamento

O processo de treinamento de uma RNN também se dá com o auxílio de um algoritmo iterativo baseado nas derivadas da função custo com respeito aos pesos sinápticos. Surge, porém, um aspecto diferente em relação às redes *feedforward* quando pensamos no cálculo do vetor gradiente: agora, a função de erro no instante t depende dos sinais gerados ao longo da estrutura da rede, mas também ao longo do tempo. Em outras palavras, a saída atual depende diretamente dos parâmetros da rede, mas também depende indiretamente deles através do vetor de estados (ou seja, dos sinais que são realimentados na rede).

A fim de explicitar a influência que os parâmetros têm ao longo do tempo e viabilizar o cálculo do vetor gradiente, a RNN é desdobrada no tempo, conforme já ilustrado anteriormente.

Uma vez construída a arquitetura desdobrada, pode-se aplicar diretamente o conceito de retropropagação do erro para calcular as derivadas da função custo com respeito aos pesos sinápticos, já levando em conta a dependência temporal dos sinais: primeiro, os padrões de entrada $\mathbf{x}(t-k), \mathbf{x}(t-k+1), \ldots, \mathbf{x}(t)$ são apresentados na entrada das respectivas estruturas e as ativações dos neurônios são propagadas pela cadeia inteira até que se obtenha as saídas no instante atual. Em seguida, calcula-se, no sentido inverso, as derivadas da função custo com relação a todos os pesos e em todos os instantes de tempo segundo a metodologia do algoritmo *backpropagation*. Finalmente, os pesos sinápticos são ajustados combinando a direção do gradiente calculado em cada instante de tempo. Esta ideia é a essência do algoritmo *backpropagation-through-time* (BPTT) (Werbos, 1990), e está ilustrada na Figura 4.

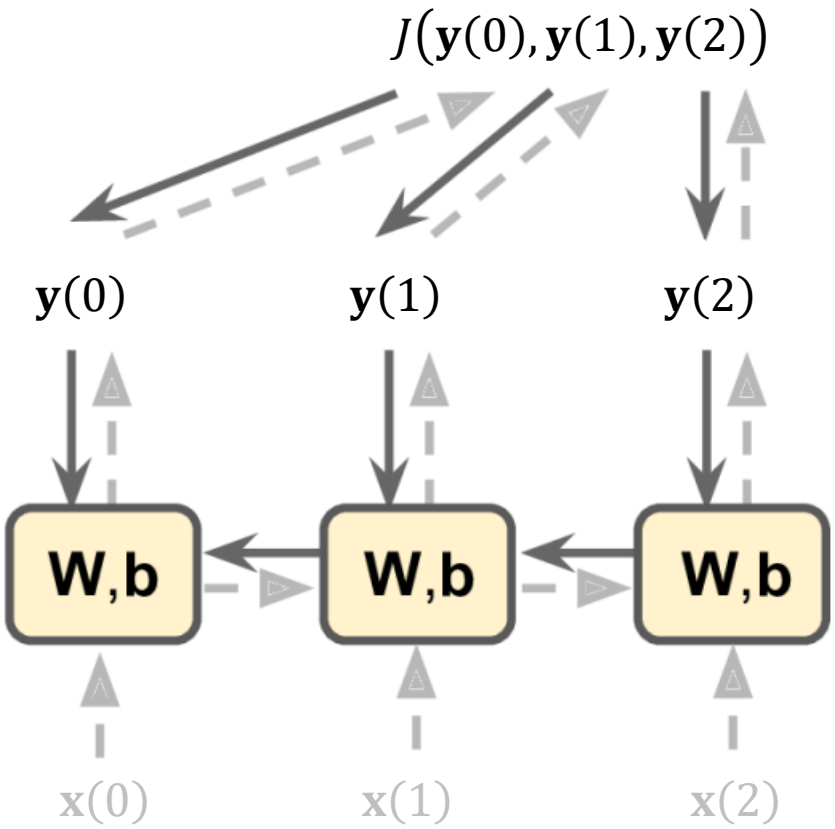

Figura 4 – Algoritmo BPTT: as setas tracejadas indicam o passo *forward*, enquanto as setas sólidas mostram o passo *backward*. Adaptada de (GÉRON, 2019).

É interessante perceber que a RNN desdobrada nada mais é do que uma rede *feedforward* mais profunda, em que as várias camadas compartilham os parâmetros.

Além disso, é bastante natural pensar em uma estrutura recorrente profunda, como mostrado na Figura 5.

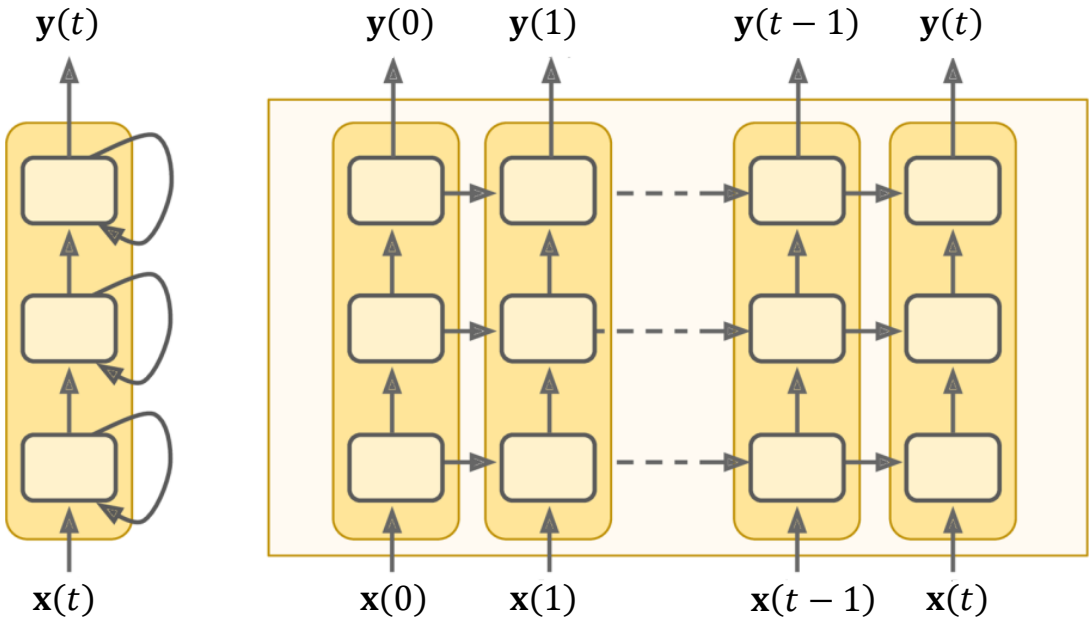

Figura 5 – *Deep* RNN e seu desdobramento no tempo. Adaptada de (GÉRON, 2019).

Não obstante, podem surgir alguns obstáculos durante a adaptação dos parâmetros de uma RNN:

- Desvanecimento do gradiente à medida que é propagado no tempo e na estrutura, o que acaba impedindo que os pesos sinápticos continuem a sofrer ajustes durante o treinamento.
- Gradientes explosivos: podem promover ajustes exagerados nos pesos, saturando neurônios e destruindo progressos anteriores já alcançados pelo método iterativo de treinamento.
    - Pequenas mudanças nos parâmetros, realizadas pelo algoritmo de treinamento, podem levar a dinâmica da rede a mudar drasticamente (*e.g.*, migrando da bacia de atração de um ponto de equilíbrio para outro), o que causa um súbito salto na medida de erro.
- Por ser um sistema dinâmico, existe sempre a preocupação com instabilidade.

Na literatura, algumas técnicas são exploradas para mitigar a ocorrência destes problemas, como *gradient clipping* e normalizações (GÉRON, 2019).

Fica evidente, portanto, que redes neurais recorrentes estão intimamente vinculadas a aplicações que envolvem contexto e informação sequencial. Com efeito, há um repertório expressivo de aplicações bem-sucedidas de RNNs em reconhecimento de fala, modelagem de linguagem, tradução de textos e rotulação de imagens.

2. Abordagens para uso de RNN

A Figura 6 mostra algumas abordagens possíveis para o emprego de RNNs.

Primeiramente, uma RNN pode receber uma sequência de entradas e produzir uma sequência de saídas. Este tipo de rede sequência-para-sequência é útil para prever séries temporais, por exemplo.

Outra possibilidade consiste em alimentar a rede com uma sequência de entradas e ignorar as saídas, exceto para o último instante de tempo. Neste caso, temos uma rede sequência-para-vetor. Como exemplo de aplicação, pense em uma rede que recebe uma sequência de palavras representando uma crítica de um filme. Neste

caso, a saída poderia ser um valor numérico simbolizando uma pontuação (*score*) relacionado ao sentimento.

Em contrapartida, poderíamos ter uma rede recebendo uma entrada única e gerando uma sequência como saída. Por exemplo, a rede poderia receber informações referentes a uma imagem, obtidas por uma rede convolucional, e produzir uma sequência de palavras para descrever o conteúdo daquela imagem.

Finalmente, temos também o caso em que uma rede sequência-para-vetor (chamada de *encoder*) gera a entrada de uma rede vetor-para-sequência (chamada de *decoder*). Este tipo de estrutura pode ser explorado para a tradução de textos de um idioma para outro.

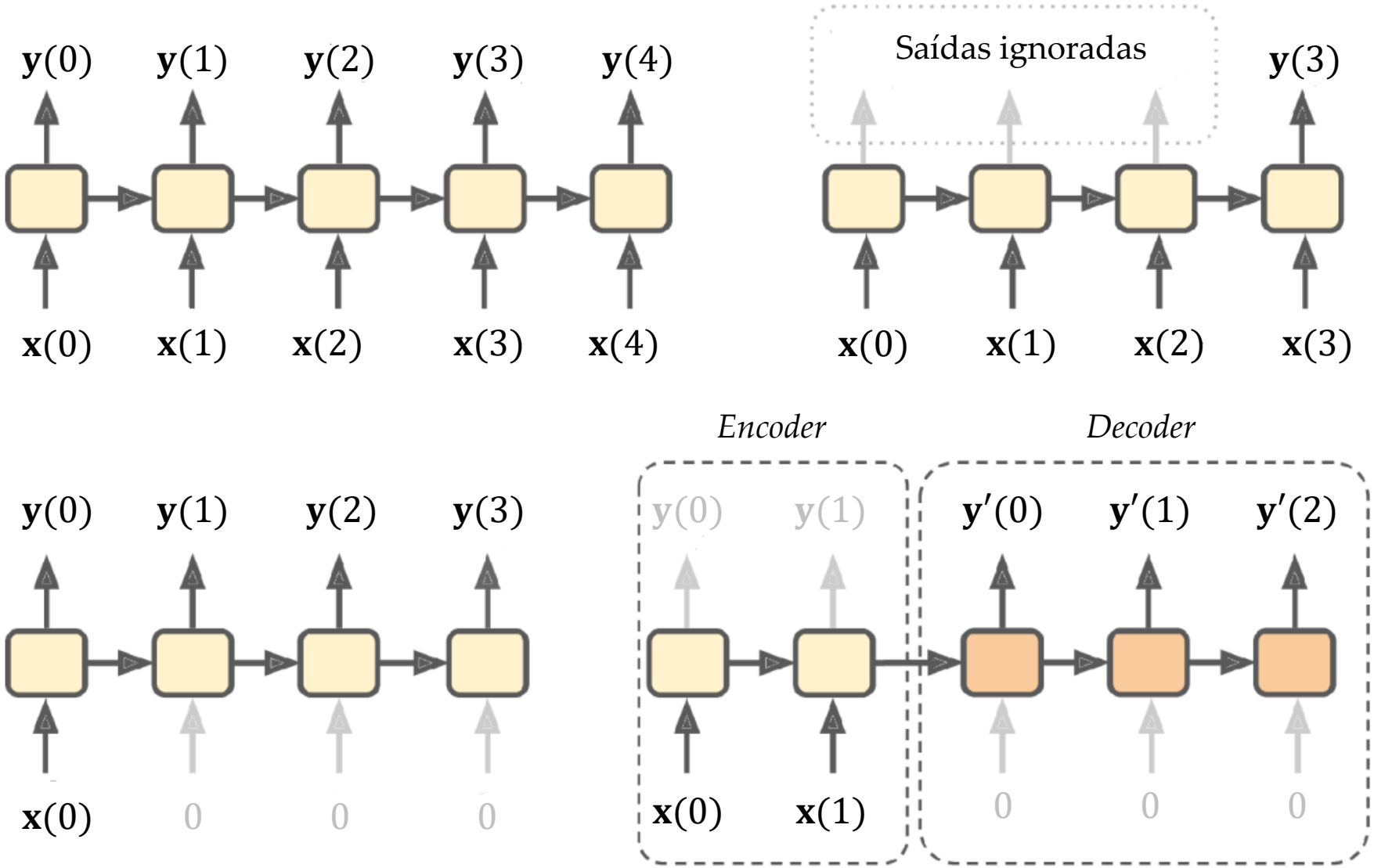

Figura 6 – Tipos de uso de RNN: sequência-para-sequência (canto esquerdo superior), sequência-para-vetor (direito superior), vetor-para-sequência (esquerdo inferior) e *encoder-decoder* (direito inferior). Adaptada de (GÉRON, 2019).

3. LSTM

Proposto em 1997, o bloco LSTM (*Long Short-Term Memory*) visa conceder a uma rede recorrente capacidade de criar simultaneamente memórias de curto e de longo prazos (HOCHREITER & SCHMIDHUBER, 1997). A Figura 7 apresenta a estrutura da célula e/ou da camada LSTM*.

A LSTM manipula um vetor de estados $\mathbf{c}(t)$, em destaque na Figura 8, o qual pode armazenar informações de longo prazo. Ao longo do treinamento, a rede aprende o que deve ser guardado nele, o que já pode ser descartado e, por fim, o que deve ser aproveitado dele para gerar a saída $\mathbf{h}(t)$.

* As figuras nesta seção estão baseadas no material em http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

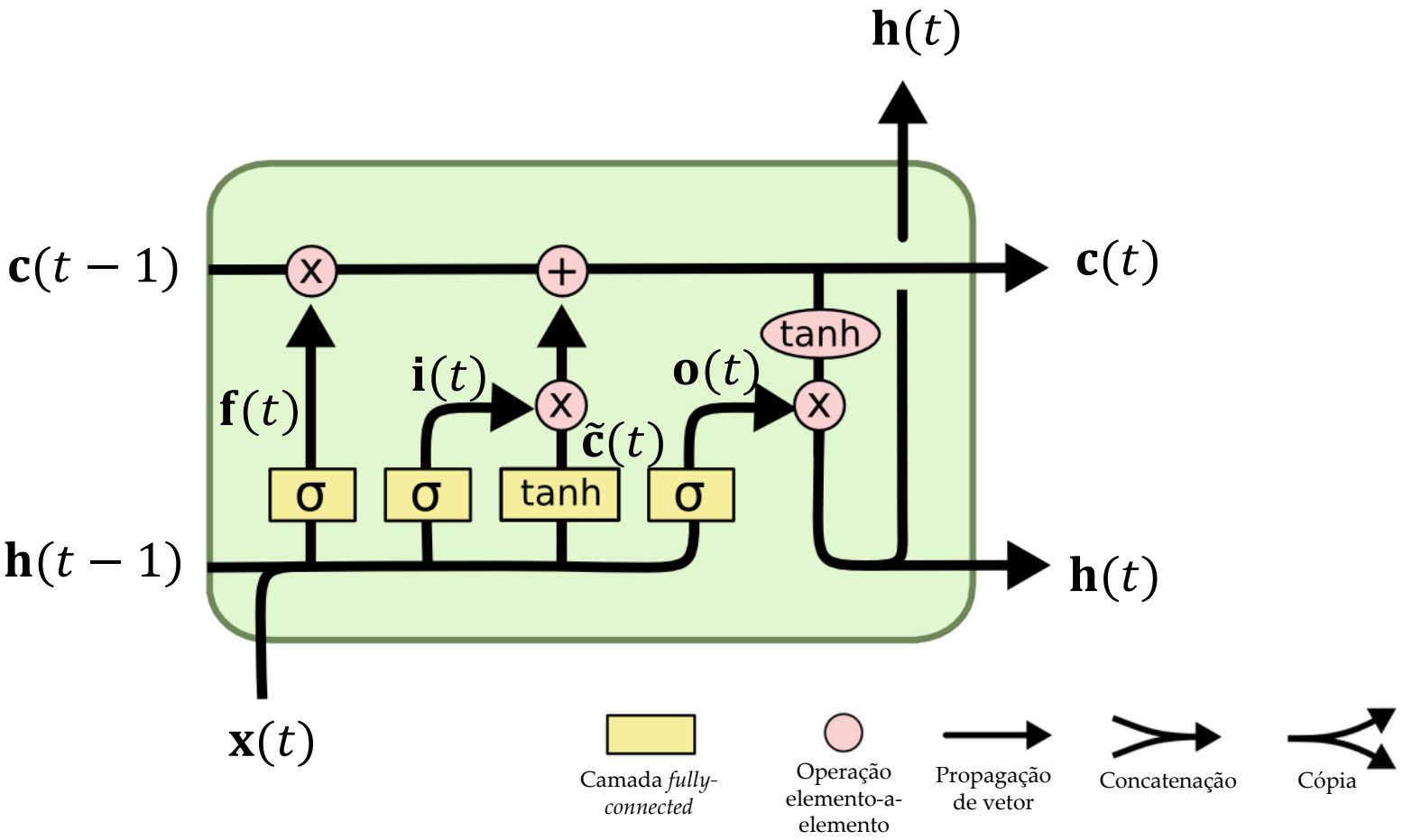

Figura 7 – Estrutura completa de uma célula/camada LSTM.

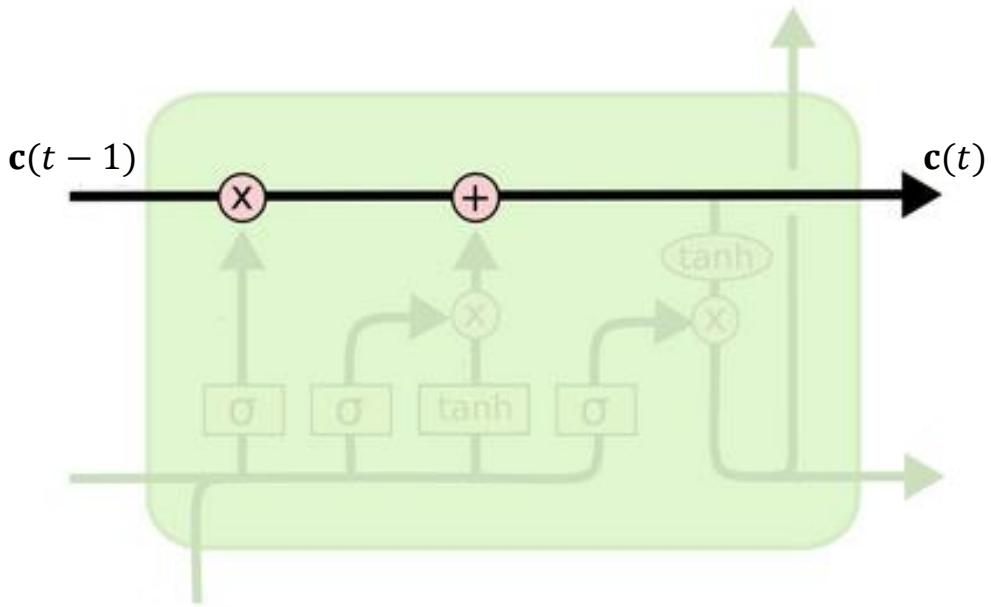

Figura 8 – Vetor de estado da LSTM.

As operações sobre o vetor de estados são controladas por três portas (em inglês, *gates*), cujo comportamento é definido durante o processo de treinamento. A Figura 9 ilustra o tipo básico de porta presente na LSTM.

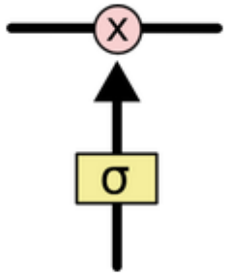

Figura 9 – Ilustração de uma porta na LSTM.

Em termos simples, cada elemento do vetor de saída de uma porta pode assumir valores no intervalo [0,1], por conta da função logística, denotada por $\sigma(\cdot)$. Assim, valores iguais a zero promovem a remoção da informação daquele canal, enquanto valores iguais a um mantêm toda a informação naquele canal.

Vejamos, agora, passo a passo, como funciona a célula LSTM.

3.1. Porta de esquecimento

O papel desta porta, destacada na Figura 10, é selecionar o que deve ser preservado para o instante t do vetor de estados $\mathbf{c}(t-1)$.

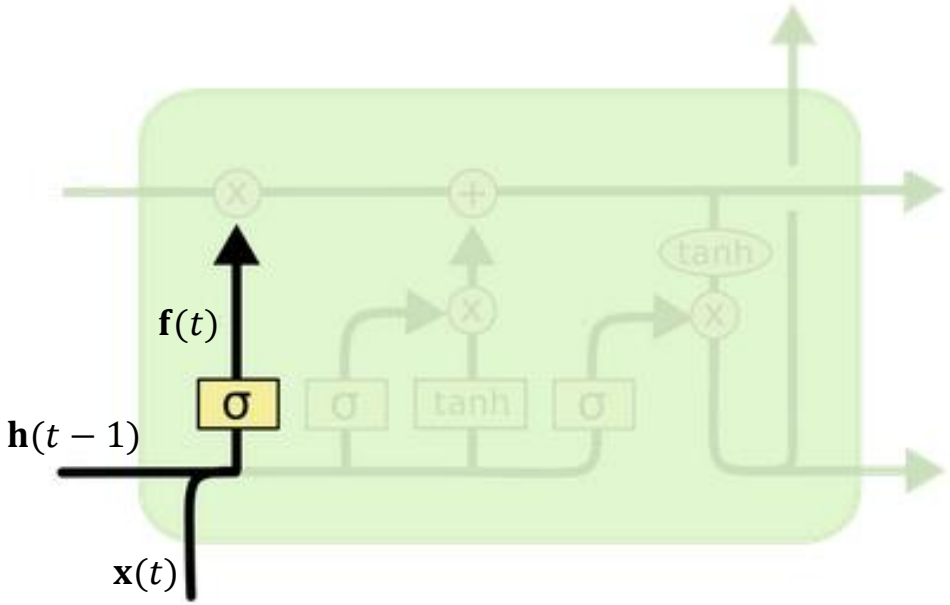

Figura 10 – *Forgetting gate*.

Dada a entrada atual $\mathbf{x}(t)$ e a saída anterior, $\mathbf{h}(t-1)$, gera-se o vetor $\mathbf{f}(t)$, que controla o comportamento da porta, através da seguinte operação:

$$\mathbf{f}(t) = \sigma\left(\mathbf{W}_f[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_f\right) \tag{4}$$

3.2. Porta de entrada

Esta porta utiliza as informações vindas da entrada atual $\mathbf{x}(t)$ e da saída passada $\mathbf{h}(t-1)$ para decidir se, e com qual intensidade, os elementos do vetor de estados serão modificados. A Figura 11 coloca em destaque este trecho da célula LSTM.

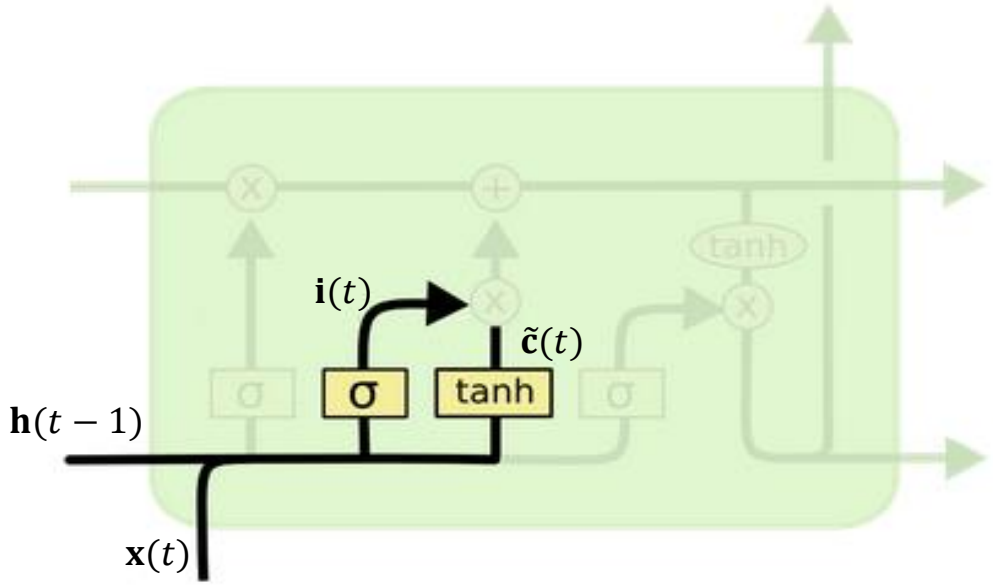

Figura 11 – *Input gate.*

O sinal de controle desta porta, denotado por $\mathbf{i}(t)$, é obtido como:

$$\mathbf{i}(t) = \sigma(\mathbf{W}_i[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_i) \tag{5}$$

As informações novas que potencialmente serão armazenadas no vetor de estados também são geradas por uma camada totalmente conectada, mas com função de ativação do tipo tangente hiperbólica:

$$\tilde{\mathbf{c}}(t) = \tanh(\mathbf{W}_C[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_C) \quad (6)$$

3.3. Atualização do vetor de estados

Com isso, o novo vetor de estados $\mathbf{c}(t)$ da célula LSTM é construído pela composição aditiva entre o que foi preservado pela porta de esquecimento e as novas informações trazidas pela porta de entrada, conforme podemos ver na Figura 12.

Ou seja,

$$\mathbf{c}(t) = \mathbf{f}(t) \otimes \mathbf{c}(t-1) + \mathbf{i}(t) \otimes \tilde{\mathbf{c}}(\mathrm{t}) \quad (7)$$

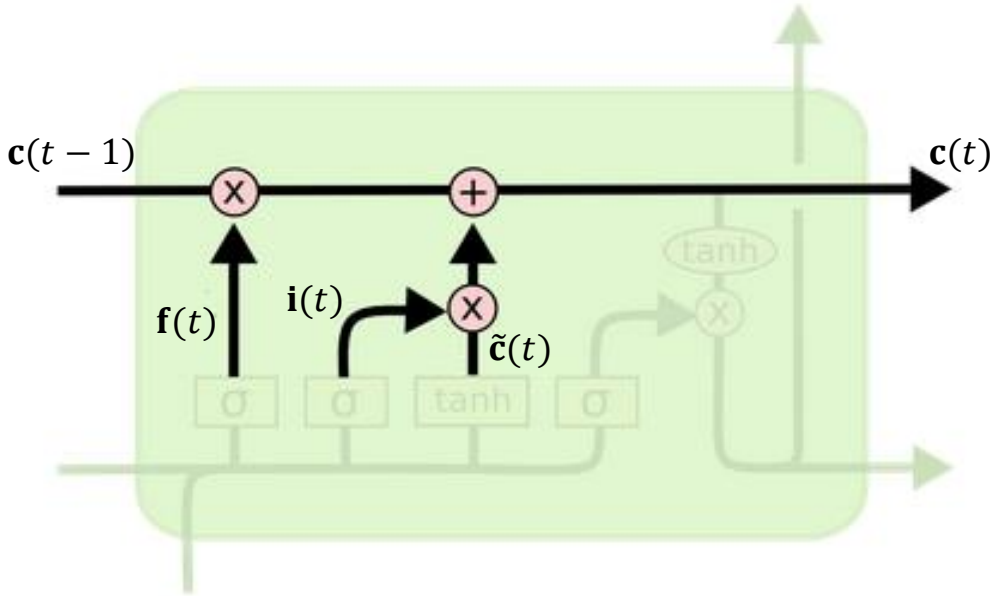

Figura 12 – Atualização do vetor de estados da célula LSTM.

3.4. Porta de saída

Finalmente, a porta de saída decide quais partes do novo vetor de estados $\mathbf{c}(t)$ devem ser capturadas na saída da célula LSTM. O detalhe é que, antes desta seleção, $\mathbf{c}(t)$ passa por uma função tangente hiperbólica, conforme ilustrado na Figura 13. Sendo assim, podemos resumir a ação da porta de saída através das seguintes equações:

$$\mathbf{o}(t) = \sigma(\mathbf{W}_o[\mathbf{h}(t-1), \mathbf{x}(t)] + \mathbf{b}_o)$$
$$\mathbf{h}(t) = \mathbf{o}(t) \otimes \tanh(\mathbf{c}(t)) \quad (8)$$

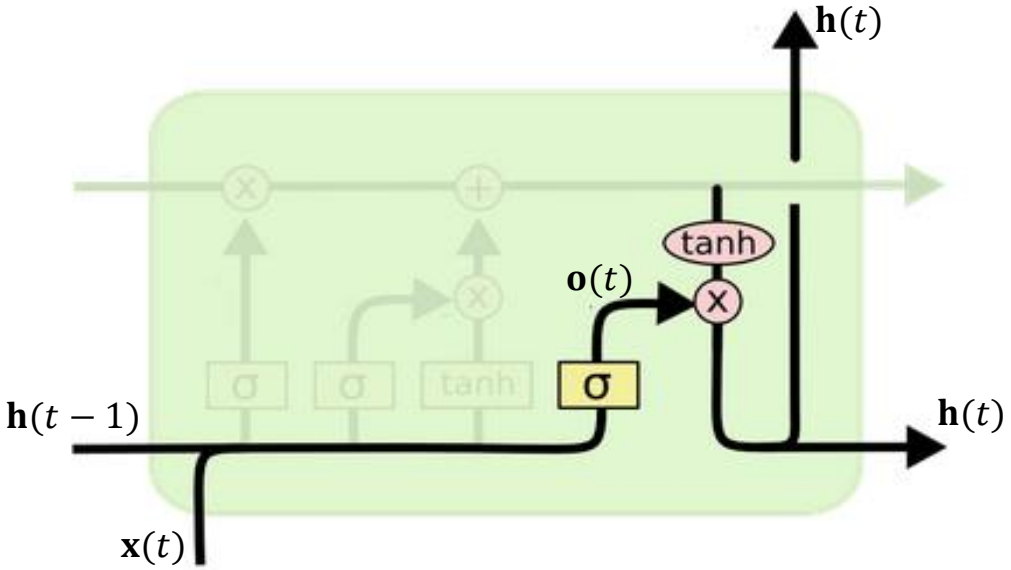

Figura 13 – Obtenção da saída da célula LSTM.

Em suma, uma célula LSTM é capaz de aprender a: (1) reconhecer uma entrada importante – este é o papel da porta de entrada –, (2) armazenar uma informação no estado de longo prazo, preservando-a pelo tempo que for necessário – este é o papel

da porta de esquecimento, (3) extrair informações do estado de longo prazo para computar a resposta da célula, o que é feito pela porta de saída.

Existem, na literatura, algumas propostas de modificações na célula LSTM (ver, por exemplo, o trabalho de GREFF ET AL., 2017). A GRU (*Gated Recurrent Unit*), proposta por Cho et al. (2014), é uma versão da LSTM com algumas simplificações (*e.g.*, o próprio vetor de estados já é a saída da célula), que se tornou bem popular por conta dos bons resultados comparativos.

4. Exemplos de aplicações

- **Geração automática de rótulos de imagens:** o sistema deve gerar um rótulo que descreva o conteúdo de uma imagem de entrada.
  - Um caminho possível consiste em utilizar uma rede profunda para detectar os objetos presentes na imagem e, em seguida, uma LSTM para converter estes identificadores em uma sentença coerente.

Figura 14 – Rótulos gerados por uma LSTM. Extraída de (VINYALS ET AL., 2015).

- **Tradução de texto:** consiste na obtenção de sentenças de texto em um idioma a partir de um trecho escrito em outro idioma. O modelo deve aprender a tradução das palavras, o contexto em que a tradução é modificada, e ser robusto a sequências de entrada e saída com tamanhos variáveis (e provavelmente distintos entre si).

Type	Sentence
Our model	Ulrich UNK , membre du conseil d' administration du constructeur automobile Audi , affirme qu' il s' agit d' une pratique courante depuis des années pour que les téléphones portables puissent être collectés avant les réunions du conseil d' administration afin qu' ils ne soient pas utilisés comme appareils d' écoute à distance .
Truth	Ulrich Hackenberg , membre du conseil d' administration du constructeur automobile Audi , déclare que la collecte des téléphones portables avant les réunions du conseil , afin qu' ils ne puissent pas être utilisés comme appareils d' écoute à distance , est une pratique courante depuis des années .
Our model	" Les téléphones cellulaires , qui sont vraiment une question , non seulement parce qu' ils pourraient potentiellement causer des interférences avec les appareils de navigation , mais nous savons , selon la FCC , qu' ils pourraient interférer avec les tours de téléphone cellulaire lorsqu' ils sont dans l' air " , dit UNK .
Truth	" Les téléphones portables sont véritablement un problème , non seulement parce qu' ils pourraient éventuellement créer des interférences avec les instruments de navigation , mais parce que nous savons , d' après la FCC , qu' ils pourraient perturber les antennes-relais de téléphonie mobile s' ils sont utilisés à bord " , a déclaré Rosenker .
Our model	Avec la crémation , il y a un " sentiment de violence contre le corps d' un être cher " , qui sera " réduit à une pile de cendres " en très peu de temps au lieu d' un processus de décomposition " qui accompagnera les étapes du deuil " .
Truth	Il y a , avec la crémation , " une violence faite au corps aimé " , qui va être " réduit à un tas de cendres " en très peu de temps , et non après un processus de décomposition , qui " accompagnerait les phases du deuil " .

Figura 15 – Exemplos de sentenças traduzidas do inglês para o francês. Extraída de (Sutskever et al., 2014).

- **Reconhecimento de emoções em sinais de fala**

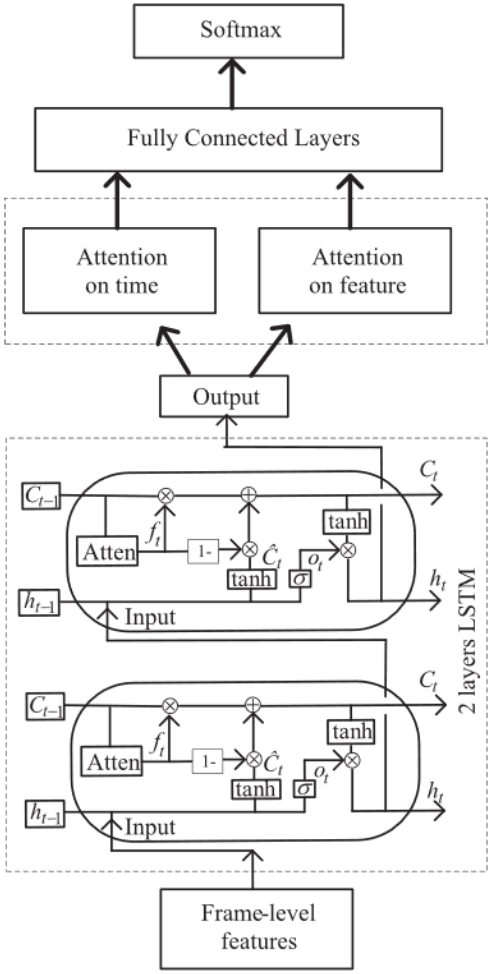

TABLE III
RESULTS OF CASIA

Models	Anger	Fear	Happy	Neutral	Sad	Surprise	UAR
LSTM	91.3%	87.1%	84.9%	95.6%	86.3%	94.8%	90.0%
LSTM-T	93.6%	91.8%	87.9%	93.0%	93.2%	91.9%	91.9%
LSTM-F	95.9%	90.1%	86.7%	95.6%	91.9%	90.8%	91.8%
LSTM-TF	87.2%	89.5%	87.3%	98.7%	95.0%	94.8%	92.0%
LSTM-at	91.9%	85.4%	86.7%	94.9%	91.3%	93.6%	90.6%
LSTM-TF-at	95.9%	88.9%	87.9%	97.5%	92.6%	94.2%	92.8%

TABLE IV
RESULTS OF ENTERFACE

Models	Anger	Disgust	Fear	Happy	Sad	Surprise	UAR
LSTM	88.4%	64.3%	76.6%	83.8%	68.9%	73.9%	75.8%
LSTM-T	93.0%	81.0%	72.3%	91.9%	82.2%	84.8%	83.8%
LSTM-F	95.4%	76.2%	78.7%	97.3%	100%	78.3%	87.3%
LSTM-TF	88.4%	85.7%	80.9%	97.3%	86.7%	84.8%	86.9%
LSTM-at	81.4%	71.4%	76.6%	94.6%	77.8%	89.1%	81.5%
LSTM-TF-at	90.7%	97.6%	76.6%	97.3%	88.9%	89.1%	89.6%

Figura 16 – Arquitetura baseada em LSTM e resultados obtidos na classificação de emoções. Extraído de (XIE ET AL., 2019).

5. Redes neurais com estados de eco

Certas dificuldades ligadas à aplicação da metodologia do gradiente a estruturas recorrentes têm atraído grande interesse para o campo de redes neurais com estados de eco (ESNs, do inglês *echo state networks*) (JAEGER, 2010; BOCCATO, 2013).

Essas redes, que possuem semelhanças com algumas ideias hoje pouco lembradas de Alan Turing (BOCCATO, 2013), possuem duas estruturas básicas: o *reservatório de dinâmicas* e a *camada de saída* (ou *readout*). A Figura 17 traz uma ilustração.

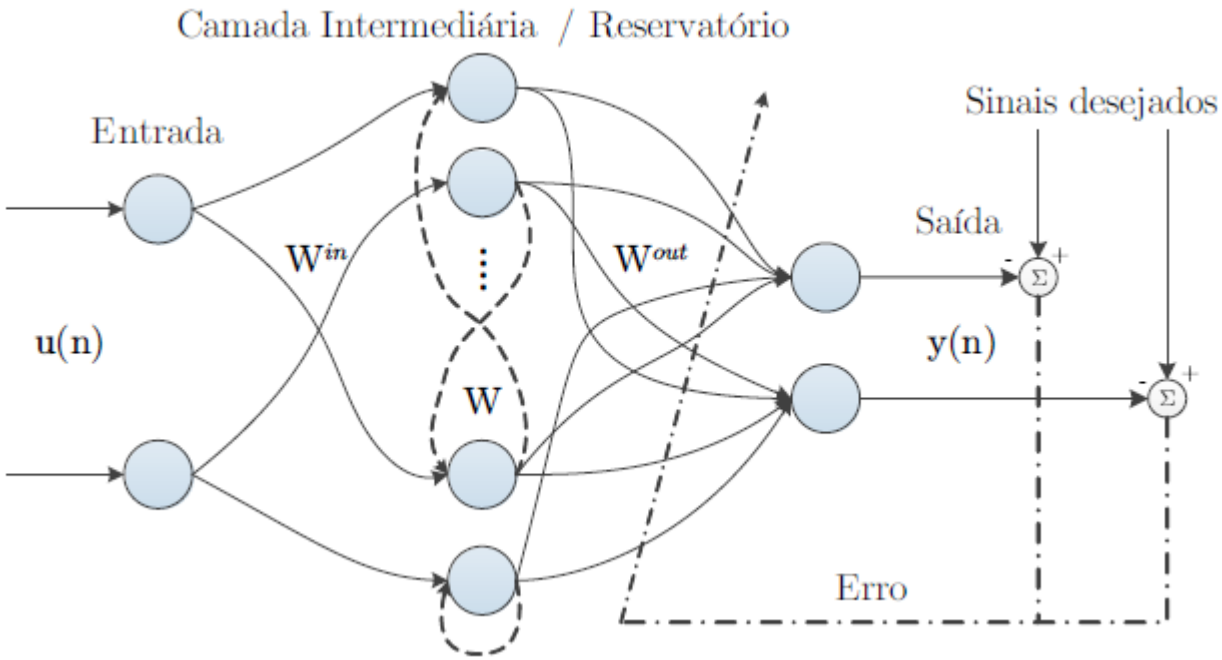

Figura 17 – Estrutura de uma Rede Neural com Estados de Eco (BOCCATO, 2013).

A camada intermediária, o reservatório de dinâmicas, contém os laços de realimentação que fazem da rede uma rede recorrente. O ponto fundamental é que, em redes com estados de eco, os pesos desse reservatório não são treinados de acordo com um sinal de referência, mas definidos *a priori* e mantidos fixos.

Isso facilita imensamente o treinamento dessa rede, pois não é necessário calcular derivadas com respeito a pesos envolvidos em laços de *feedback*. Resta apenas

otimizar os pesos da camada de saída (*readout*), que engendram um modelo linear nos parâmetros e, destarte, podem ser obtidos até mesmo em forma fechada (por mínimos quadrados).

As equações que regem a ESN são expressas, usualmente, como†:

$$\begin{aligned} \mathbf{x}(n) &= f\left(\mathbf{W}\mathbf{x}(n-1) + \mathbf{W}^{\text{in}}\mathbf{u}(n)\right) \\ \mathbf{y}(n) &= \mathbf{W}^{\text{out}}\mathbf{x}(n) \end{aligned} \tag{9}$$

Há algumas regras e princípios para escolha das matrizes de pesos, especialmente da matriz do reservatório $\mathbf{W}$. Na definição dessa matriz, é preciso levar em conta *inter alia* a propriedade de estados de eco (JAEGER, 2010).

† Os termos de *bias* foram omitidos por simplicidade. Eventuais não-linearidades da camada de saída, bem como *feedback* direto dessa camada, foram desconsiderados.

6. Referências bibliográficas

BOCCATO, L. Novas Propostas e Aplicações de Redes Neurais com Estados de Eco. Tese de Doutorado. Faculdade de Engenharia Elétrica e de Computação, UNICAMP, 2013.

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.

GÉRON, A., **Hands-on Machine Learning with Scikit-Learn, Keras & Tensorflow**, O'Reilly Media, 2ª ed., 2019.

GOODFELLOW, I., BENGIO, Y., COURVILLE, A., **Deep Learning**, MIT Press, 2016.

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.

HOCHREITER, S., SCHMIDHUBER, J., "Long Short-Term Memory", *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

JAEGER, H., The "Echo State" Approach to Analysing and Training Recurrent Neural Networks – with an Erratum Note, Technical Report, 2010.

SUTSKEVER, I., VINYALS, O., LE, Q. V., "Sequence to Sequence Learning with Neural Networks", Proceedings of the 27th International Conference on Neural Information Processing Systems, vol. 2, pp. 3104-3112, 2014.

VINYALS, O., TOSHEV, A., BENGIO, S., ERHAN, D., "Show and Tell: A Neural Image Caption Generator", Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CPVR), pp. 3156-3164, 2015.

VON ZUBEN, F. J., **Notas de Aulas do Curso "Redes Neurais" (IA353),** disponíveis em http://www.dca.fee.unicamp.br/~vonzuben/courses/ia353.html

WERBOS, P. J., "Backpropagation through time: what it does and how to do it". *Proceedings of the IEEE*, vol. 78, pp. 1550-1560, 1990.

XIE, Y., LIANG, R., LIANG, Z., HUANG, C., ZOU, C., SCHULLER, B., "Speech Emotion Classification Using Attention-Based LSTM", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 11, pp. 1675-1685, 2019.