TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ**

**MÔN HỌC MÁY**

**LÝ THUYẾT MẠNG RECURRENT NEURAL NETWORK**

*Người hướng dẫn*: **Thầy Lê Anh Cường**

*Người thực hiện*: **TRẦN VĂN AN– 518H0127**

**LÊ HOÀNG LONG – 518H0035**

Lớp **: 18H50205**

Khoá  **: 22**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ**

**MÔN HỌC MÁY**

**LÝ THUYẾT MẠNG RECURRENT NEURAL NETWORK**

*Người hướng dẫn*: **Thầy Lê Anh Cường**

*Người thực hiện*: **TRẦN VĂN AN– 518H0127**

**LÊ HOÀNG LONG – 518H0035**

Lớp **: 18H50205**

Khoá  **: 22**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020

LỜI CẢM ƠN

Chúng em cảm ơn thầy Lê Anh Cường đã trực tiếp giảng dạy bộ môn này.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của TS Nguyễn Văn A;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Văn B*

*Trần Văn C*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Trình bày tóm tắt vấn đề nghiên cứu, các hướng tiếp cận, cách giải quyết vấn đề và một số kết quả đạt được, những phát hiện cơ bản trong vòng 1 -2 trang.

MỤC LỤC

Contents

[CHƯƠNG 1: GIỚI THIỆU 2](#_Toc71416593)

[1.1 Giới thiệu tổng quan. 2](#_Toc71416594)

[1.2 MẠNG RNN(RECURRENT NEURAL NETWORK) là gì? 3](#_Toc71416595)

[CHƯƠNG 2: PHÂN LOẠI KIẾN TRÚC RNN 6](#_Toc71416596)

[2.1 Dữ liệu dạng sequence 6](#_Toc71416597)

[2.3 Phân loại 6](#_Toc71416598)

[2.3.1 One-to-one 7](#_Toc71416599)

[2.3.2 One-to-many 7](#_Toc71416600)

[2.3.3 Many-to-one 8](#_Toc71416601)

[2.3.4 Many-to-many 9](#_Toc71416602)

[2.4 Loss function 9](#_Toc71416603)

[2.5 Backprogation through time 10](#_Toc71416604)

[CHƯƠNG 3: LONG SHORT-TERM MEMORY 11](#_Toc71416605)

[3.1 Giới thiệu 11](#_Toc71416606)

[3.2 Cách LSTM chống vanishing gradient 13](#_Toc71416607)

[CHƯƠNG 4: TÀI LIỆU THAM KHẢO 14](#_Toc71416608)

[Link lý thuyết: 14](#_Toc71416609)

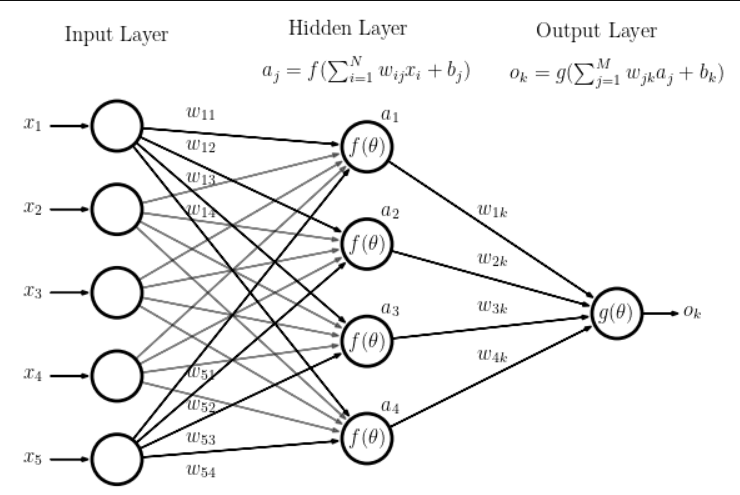
[Link demo: 14](#_Toc71416610)

CHƯƠNG 1: GIỚI THIỆU

* 1. Giới thiệu tổng quan.

Mạng nơ-ron hồi quy ( RNN - Recurrent Neural Network) là sử dụng chuỗi các thông tin. Trong các mạng nơ-ron truyền thống tất cả các đầu vào và cả đầu ra là độc lập với nhau. Tức là chúng không liên kết thành chuỗi với nhau. Nhưng các mô hình này không phù hợp trong rất nhiều bài toán. Ví dụ, nếu muốn đoán từ tiếp theo có thể xuất hiện trong một câu thì ta cũng cần biết các từ trước đó xuất hiện lần lượt thế nào thì rất là khó. RNN được gọi là hồi quy (Recurrent) bởi lẽ chúng thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Nói cách khác, RNN có khả năng nhớ các thông tin được tính toán trước đó. Trên lý thuyết, RNN có thể sử dụng được thông tin của một văn bản rất dài, tuy nhiên thực tế thì nó chỉ có thể nhớ được một vài bước trước đó. Về cơ bản một mạng RNN có dạng như sau:

Để có thể hiểu rõ về RNN, trước tiên chúng ta cùng nhìn lại mô hình Neural Network dưới đây:



Như đã biết thì Neural Network bao gồm 3 phần chính là Input layer, Hidden layer và Output layer, ta có thể thấy là đầu vào và đầu ra của mạng neuron này là độc lập với nhau. Như vậy mô hình này không phù hợp với những bài toán dạng chuỗi như mô tả, hoàn thành câu, ... vì những dự đoán tiếp theo như từ tiếp theo phụ thuộc vào vị trí của nó trong câu và những từ đằng trước nó.Và như vậy RNN ra đời với ý tưởng chính là sử dụng một bộ nhớ để lưu lại thông tin từ từ những bước tính toán xử lý trước để dựa vào nó có thể đưa ra dự đoán chính xác nhất cho bước dự đoán hiện tại.

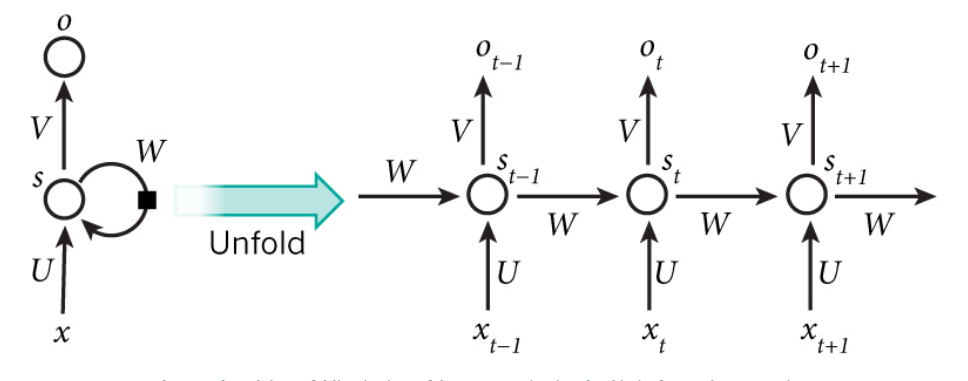
1.2 MẠNG RNN(RECURRENT NEURAL NETWORK) là gì?

Mạng nơ-ron tuần hoàn (RNN) là một lớp mạng nơ-ron nhân tạo trong đó các kết nối giữa các nút tạo thành một đồ thị có hướng dọc theo một trình tự thời gian. Điều này cho phép nó thể hiện hành vi động tạm thời. Bắt nguồn từ mạng nơ-ron truyền thẳng, RNN có thể sử dụng trạng thái bên trong (bộ nhớ) của chúng để xử lý chuỗi đầu vào có độ dài thay đổi. Ý tưởng chính của RNN (Recurrent Neural Network) là sử dụng chuỗi các thông tin, trong đó đầu ra từ bước trước được cung cấp làm đầu vào cho bước hiện tại. Khác với các mạng nơron sâu truyền thống giả định rằng đầu vào và đầu ra là độc lập với nhau, đầu ra của mạng RNN phụ thuộc vào các phần tử trước đó trong chuỗi.Và các dạng dữ liệu mà RNN sử dụng là dữ liệu tuần tự(sequential) và dữ liệu thời gian(time-series).

Tính năng chính và quan trọng nhất của RNN là “**Hidden State**”, nó ghi nhớ một số thông tin về một chuỗi.

RNN có 1 phần gọi là “**memory**”, trong đó ghi nhớ tất cả thông tin về những gì đã được tính toán. Nó sử dụng các tham số giống nhau cho mỗi đầu vào vì nó thực hiện cùng một nhiệm vụ trên tất cả các đầu vào hoặc các lớp ẩn để tạo ra đầu ra. Điều này làm giảm độ phức tạp của các tham số.

Về cơ bản thì RNN có dạng như sau:



Mô hình trên nội dung triển khai mô tả của một RNN. Triển khai ở đây có thể hiểu đơn giản là chúng ta vẽ ra một mạng nơ-ron chuỗi tuần tự. Ví dụ ta có một câu gồm 5 chữ "Đẹp trai lắm gái theo", thì mạng nơ-ron được triển khai sẽ gồm 5 tầng nơ-ron tương ứng với mỗi chữ một tầng. Lúc đó bên chức năng tính toán trong RNN được thực hiện như sau:

* Xt là đầu vào tại bước t. Ví dụ x1 là một vector tương ứng với từ thứ 2 của câu(trai).
* St là trạng thái ẩn tại t . Nó chính là bộ nhớ của mạng.st được tính toán dựa trên các trạng thái ẩn phía trước và đầu vào tại bước đó: st = f.(U.xt+W.st-1).Hàm F thường là một hàm phi tuyến tính(activation function) .Để làm phép toán cho phần tử đầu tiên ta cần khởi tạo thêm s-1,thường giá trị khởi tạo gán bằng 0.
* 0t là đầu vào tại bước t. Ví dụ, ta muốn dự đoán từ tiếp theo có thể xuất hiện trong câu thì ot chính là 1 vector xác xuất các từ trong danh sách từ vựng của ta: ot = softmax(V.st).

CHƯƠNG 2: PHÂN LOẠI KIẾN TRÚC RNN

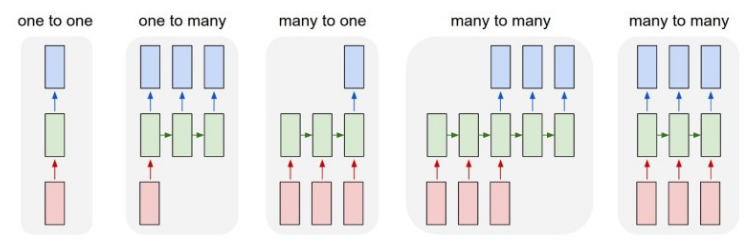
Deep learning có 2 mô hình lớn là Convolutional Neural Network (CNN) cho bài toán có input là ảnh và Recurrent neural network (RNN) cho bài toán dữ liệu dạng chuỗi (sequence). Trong bài này chúng ta tập trung nói về RNN.

2.1 Dữ liệu dạng sequence

Dữ liệu có thứ tự ví dụ như các ảnh tách từ một video nào đó được gọi là sequence, time-series data.

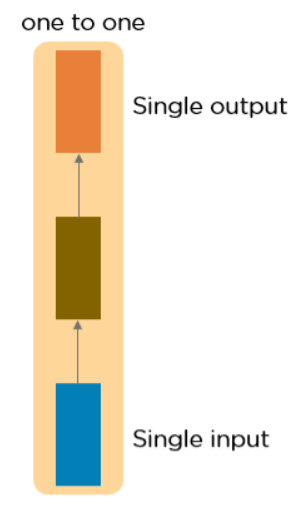
Hoặc trong bài toán dự đoán đột quỵ tim cho bệnh nhân bằng các dữ liệu tim mạch khám trước đó. Input là dữ liệu của những lần khám trước đó, ví dụ i1 là lần khám tháng 1, i2 là lần khám tháng 2,… i8 là lần khám tháng 8. (i1,i2,..i8) được gọi là sequence data. RNN sẽ học từ input và dự đoán xem bệnh nhân có bị đột quy tim hay không.

Ví dụ khác là trong bài toán dịch tự động với input là 1 câu, ví dụ “tôi yêu Việt Nam” thì vị trí các từ và sự xắp xếp cực kì quan trọng đến nghĩa của câu và dữ liệu input các từ [‘tôi’, ‘yêu’, ‘việt’, ‘nam’] được gọi là sequence data.

2.3 Phân loại

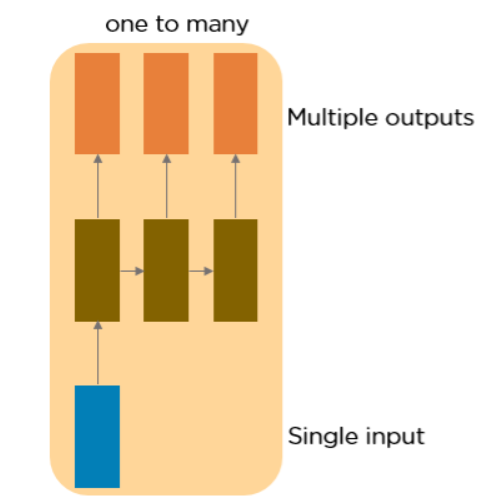
2.3.1 One-to-one

Đây là kiến trúc phổ biến nhất của RNN. Nó còn được gọi là vainila network. Kiến trúc này cung cấp 1 đầu vào và 1 đầu ra duy nhất. Khi Tx​=Ty​=1



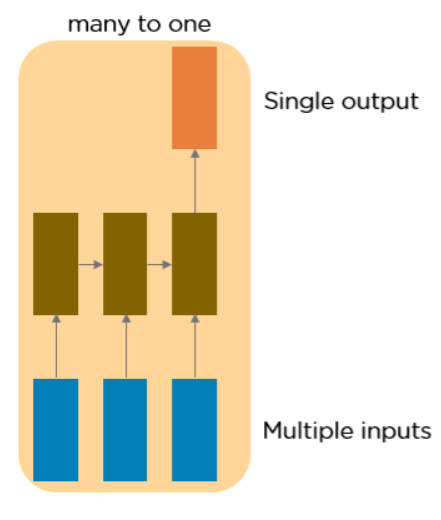
2.3.2 One-to-many

Nó có 1 đầu vào và nhiều đầu ra. Nó được sử dụng trong các tình huống cần nhiều đầu ra cho một đầu vào duy nhất. Nó có thể được sử dụng trong nhiều lĩnh vực khác nhau như trong ngành công nghiệp âm nhạc. Khi Tx​=1,Ty​>1



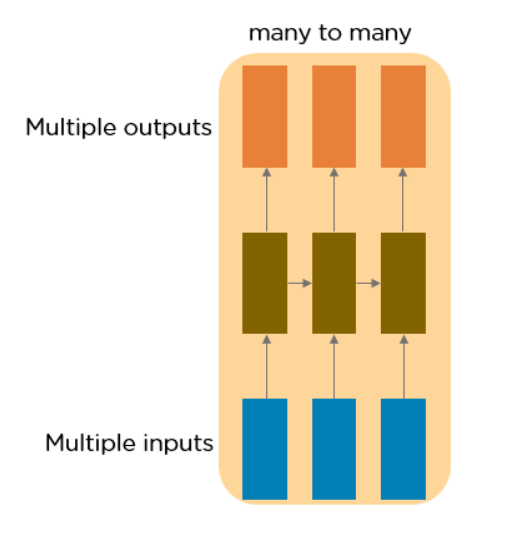
2.3.3 Many-to-one

RNN này nhận một chuỗi các đầu vào và tạo ra một đầu ra duy nhất. Phân tích cảm xúc là một ví dụ điển hình của loại mạng này, trong đó một câu nhất định có thể được phân loại là thể hiện tình cảm tích cực hoặc tiêu cực.

Khi Tx​>1,Ty​=1

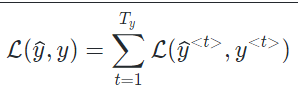
2.3.4 Many-to-many

RNN này nhận một chuỗi các đầu vào và tạo ra một chuỗi các đầu ra. Dịch máy là một trong những ví dụ.Khi Tx​=Ty​



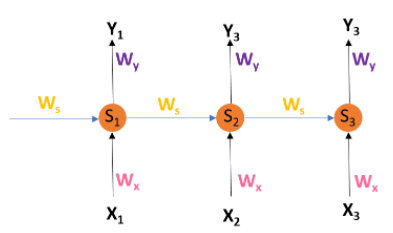
2.4 Loss function

Trong trường hợp mạng nơron lặp lại, hàm mất mát L của tất cả các bước thời gian được xác định dựa trên tổn thất tại mỗi bước thời gian như sau:



2.5 Backprogation through time

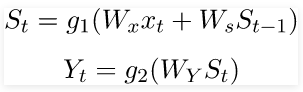
BPTT là thuật toán được sử dụng để cập nhật các trọng số trong RNN. Ta không đào tạo hệ thống một cách độc lập tại một thời điểm cụ thể "t". Ta huấn luyện nó vào một thời điểm cụ thể "t" cũng như tất cả những gì đã xảy ra trước thời điểm "t" như t-1, t-2, t-3. Mục đích của thuật toán Backpropagation là giảm thiểu lỗi của các kết quả đầu ra của mạng.



S1, S2, S3 lần lượt là các trạng thái ẩn hoặc đơn vị bộ nhớ tại thời điểm t1, t2, t3 và Ws là ma trận trọng số liên kết với nó.

X1, X2, X3 lần lượt là các đầu vào tại thời điểm t1, t2, t3 và Wx là ma trận trọng số liên quan đến nó.

Y1, Y2, Y3 lần lượt là đầu ra tại thời điểm t1, t2, t3 và Wy là ma trận trọng số liên quan đến nó.

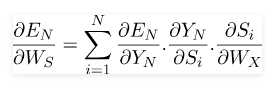
Ta có công thức:

Trong đó g1 và g2 là hàm activation và Hàm error có công thức như sau:



Để thực hiện backpropogation, chúng ta phải điều chỉnh trọng số liên quan đến đầu vào, tầng ẩn và đầu ra.

Công thức tổng quát là :



CHƯƠNG 3: LONG SHORT-TERM MEMORY

3.1 Giới thiệu

Mặc dù Mạng RNN cơ bản khá là hiệu quả, nhưng mọi thứ đều có hạn chế của nó, đối với mạng RNN hhương pháp Truyền ngược xuyên thời gian (BPTT) này có thể được sử dụng tới một số bước thời gian giới hạn như 8 hoặc 10. Nếu chúng ta quay lại lan truyền xa hơn, gradient sẽ trở nên quá nhỏ hoặc sấp sĩ 0. Sự cố này được gọi là sự cố "Vanishing gradient",đó là khi lượng thông tin bị giảm dần theo thời gian.

Một trong những giải pháp nổi tiếng cho vấn đề này là sử dụng các ô được gọi là Long short-term memory (viết tắt là LSTM) thay vì các ô RNN truyền thống.

Về khái niệm thì LSTM sẽ cố gắng nhớ hết tất cả các thông tin trước đó và bị quên những dữ liệu không liên quan. Một LSTM bao gồm bốn cổng khác nhau cho các mục đích khác nhau:

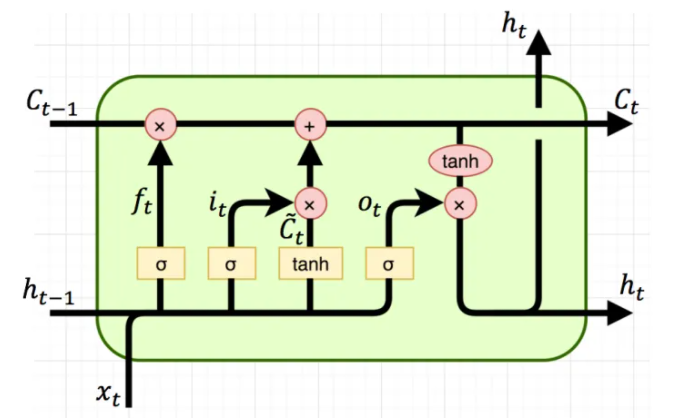
1.Forget Gate(f): Nó quyết định việc quên dữ liệu trước đó ở mức độ nào.

2. Input Gate(i): Nó xác định mức độ thông tin được ghi vào Internal Cell State

3.Input Modulation Gate(g): Nó thường được coi là một phần phụ của cổng đầu vào. Nó được sử dụng để điều chỉnh thông tin mà Cổng vào sẽ ghi vào **Internal State Cell** trong bằng cách thêm tính không tuyến tính vào thông tin và làm cho thông tin có **Zero-mean**. Điều này được thực hiện để giảm thời gian học tập vì đầu vào Zero-mean có sự hội tụ nhanh hơn. Mặc dù các hành động của cổng này ít quan trọng hơn các cổng khác và thường được coi là một khái niệm cung cấp sự khéo léo, nhưng thực tế tốt là đưa cổng này vào cấu trúc của đơn vị LSTM

4. Output gate(o): Nó xác định đầu ra (Trạng thái ẩn tiếp theo) sẽ tạo ra từ Trạng thái ô bên trong hiện tại.

Mô hình của LSTM:

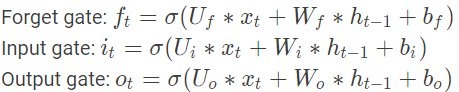


Output: ct,ht ta gọi là cell state, hidden state.

Input: ct-1,ht-1,xt. Trong đó xt là input ở state thứ t của model. Ct-1 ,ht-1 là output của layer trước. **h** đóng vai trò khá giống như **s** ở RNN,trong khi đó **c** là điểm mới của LSTM.

Ở biểu đồ trên, kí hiệu σ, tanh là dùng sigma,tanh function. Phép nhân là element-wise-multiplication,phép cộng là cộng ma trận.

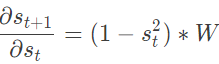
Ft, it,ot tương ứng với forget state, input gate, output gate.



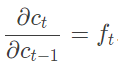
Nhận xét: 0< ft,it,ot <1;bf,bi,bo là các hệ số bias,hệ số W,U giống trong RNN.

3.2 Cách LSTM chống vanishing gradient

Ta cũng áp dụng thuật toán BPTT cho LSTM tương tự như RNN.

Thành phần chính gây ra vanishing gradient trong RNN là

trong đó st, W <1. Sau một thời gian nhất định bắt đầu nhận các giá trị lớn hơn 1 hoặc nhỏ hơn 1 nhưng luôn trong cùng một phạm vi.

Tương tự như vậy trong LSTM ta quan tâm đến  do nó không có mẫu cố định và có thể nhận bất kỳ giá trị dương nào tại bất kỳ time step nào. Do đó, không chắc chắn rằng đối với một số bước thời gian vô hạn, số hạng sẽ hội tụ về 0 hoặc phân kỳ hoàn toàn. Nếu gradient bắt đầu hội tụ về 0, thì trọng số của các cổng có thể được điều chỉnh tương ứng để đưa nó về gần 1. Vì trong giai đoạn huấn luyện, mạng chỉ điều chỉnh các trọng số này, do đó nó học được khi nào để gradient hội tụ về 0 và khi nào thì bảo quản nó => tránh được vanishing gradient.

CHƯƠNG 4: TÀI LIỆU THAM KHẢO

Link lý thuyết:

<https://nttuan8.com/bai-13-recurrent-neural-network/>

<https://nttuan8.com/bai-14-long-short-term-memory-lstm/>

<https://nttuan8.com/bai-14-long-short-term-memory-lstm/>

<https://dominhhai.github.io/vi/2017/10/what-is-rnn/>

Link demo:

<https://www.tensorflow.org/tutorials/structured_data/time_series?fbclid=IwAR3B9d884cpQ-n7NZpa_F17QtIiewti_4V1IeioWe43zzshHZt8m-9uaKTs#recurrent_neural_network>

<https://machinelearningmastery.com/how-to-make-classification-and->regression-predictions-for-deep-learning-models-in-keras/

<https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/>

<https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense>

<https://stats.stackexchange.com/questions/257996/what-is-maxnorm-constraint-how-is-it-useful-in-convolutional-neural-networks>

<https://keras.io/api/layers/regularization_layers/dropout/#:~:text=Dropout%20class&text=The%20Dropout%20layer%20randomly%20sets,over%20all%20inputs%20is%20unchanged>.

<https://www.tensorflow.org/api_docs/python/tf/keras/Modelhttps://www.tensorflow.org/api_docs/python/tf/keras/callbacks/EarlyStopping>

<https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn>

<https://learnopencv.com/image-classification-using-feedforward-neural-network-in-keras/>