TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**DỰ ÁN CUỐI KÌ**

**MÔN NHẬP MÔN HỌC MÁY**

**MẠNG RECURRENT NEURAL NETWORK (RNN)**

*Người hướng dẫn*: **TS LÊ ANH CƯỜNG**

*Người thực hiện*: **TRẦN NGỌC THÁI SƠN – 518H0262**

**NGUYỄN HỮU TÀI – 518H0558**

Lớp **: 18H50205**

Khoá  **: 22**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**DỰ ÁN CUỐI KÌ**

**MÔN NHẬP MÔN HỌC MÁY**

**MẠNG RECURRENT NEURAL NETWORK (RNN)**

*Người hướng dẫn*: **TS LÊ ANH CƯỜNG**

*Người thực hiện*: **TRẦN NGỌC THÁI SƠN – 518H0262**

**NGUYỄN HỮU TÀI – 518H0558**

Lớp **: 18H50205**

Khoá  **: 22**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

LỜI CẢM ƠN

Cảm ơn thầy Lê Anh Cường đã hỗ trợ và giảng dạy cho em trong suốt học kỳ vừa qua. Nhờ những kiến thức mà thầy truyền đạt mà chúng em đã có thể hoàn thành dự án cuối kỳ một cách tốt nhất. Em xin chân thành cảm ơn thầy.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của TS Lê Anh Cường;. 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)*

*Trần Ngọc Thái Sơn*

*Nguyễn Hữu Tài*

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

Đây là sơ lược về lý thuyết mạng Recurrent Neural Network (RNN) và Long Short Term Memory (LSTM)

Mục lục

[**DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT** 8](#_Toc71403417)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 9](#_Toc71403418)

[**CHƯƠNG 1** 10](#_Toc71403419)

[Recurrent Neural Network là gì? 10](#_Toc71403420)

[**1.1. Lý do sử dụng mạng RNN:** 10](#_Toc71403421)

[**1.2. Dữ liệu dạng sequence:** 10](#_Toc71403422)

[**1.3. Phân loại bài toán RNN:** 11](#_Toc71403423)

[**1.4. Ứng dụng bài toán RNN:** 11](#_Toc71403424)

[**CHƯƠNG 2** 13](#_Toc71403425)

[Mô hình bài toán RNN 13](#_Toc71403426)

[**2.1. Mô hình RNN:** 13](#_Toc71403427)

[**2.2. Loss function:** 14](#_Toc71403428)

[**CHƯƠNG 3** 17](#_Toc71403429)

[LSTM 17](#_Toc71403430)

[**CHƯƠNG 4** 20](#_Toc71403431)

[Màn Hình Kết Quả Code 20](#_Toc71403432)

**DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT**

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**CHƯƠNG 1**

## Recurrent Neural Network là gì?

* 1. **Lý do sử dụng mạng RNN:**

Khi xử lý video ta hay gặp khái niệm FPS (frame per second) tức là bao nhiêu frame (ảnh) mỗi giây. Ví dụ 1 FPS với video 30s tức là lấy ra từ video 30 ảnh, mỗi giây một ảnh để xử lý.

Ta dùng 1 FPS cho video input ở bài toán trên, tức là lấy ra 30 ảnh từ video, ảnh 1 ở giây 1, ảnh 2 ở giây 2,… ảnh 30 ở giây 30. Bây giờ input là 30 ảnh: ảnh 1, ảnh 2,… ảnh 30 và output là phân loại hành động. Nhận xét:

* Các ảnh có **thứ tự** ví dụ ảnh 1 xẩy ra trước ảnh 2, ảnh 2 xảy ra trước ảnh 3,… Nếu ta đảo lộn các ảnh thì có thể thay đổi nội dung của video. Ví dụ: nội dung video là cảnh bắn nhau, thứ tự đúng là A bắn trúng người B và B chết, nếu ta đảo thứ tự ảnh thành người B chết xong A mới bắn thì rõ ràng bây giờ A không phải là kẻ giết người => nội dung video bị thay đổi.
* Ta có thể dùng CNN để phân loại 1 ảnh trong 30 ảnh trên, nhưng rõ ràng là 1 ảnh không thể mô tả được nội dung của cả video. Ví dụ: Cảnh người cướp điện thoại, nếu ta chỉ dùng 1 ảnh là người đấy cầm điện thoại lúc cướp xong thì ta không thể biết được cả hành động cướp.

=> Cần một mô hình mới có thể giải quyết được bài toán với input là sequence (chuỗi ảnh 1->30) => RNN ra đời.

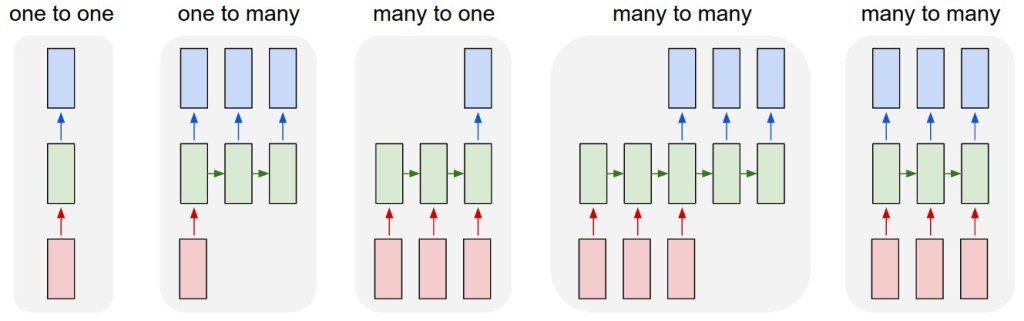
* 1. **Dữ liệu dạng sequence:**

- Dữ liệu có thứ tự như các ảnh tách từ video ở trên được gọi là sequence, time-series data.

- 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. **Trong bài toán xử lý ngôn ngữ (NLP) thì không thể xử lý cả câu được và người ta tách ra từng từ làm input, giống như trong video người ta tách ra các ảnh (frame) làm input.**

* 1. **Phân loại bài toán RNN:**



Các dạng bài toán RNN

**- One to one**: mẫu bài toán cho Neural Network (NN) và Convolutional Neural Network (CNN), 1 input và 1 output, ví dụ với CNN input là ảnh và output là ảnh được segment.

**- One to many**: bài toán có 1 input nhưng nhiều output, ví dụ: bài toán caption cho ảnh, input là 1 ảnh nhưng output là nhiều chữ mô tả cho ảnh đấy, dưới dạng một câu.

**- Many to one**: bài toán có nhiều input nhưng chỉ có 1 output, ví dụ bài toán phân loại hành động trong video, input là nhiều ảnh (frame) tách ra từ video, ouptut là hành động trong video

**- Many to many**: bài toán có nhiều input và nhiều output, ví dụ bài toán dịch từ tiếng anh sang tiếng việt, input là 1 câu gồm nhiều chữ: “I love Vietnam” và output cũng là 1 câu gồm nhiều chữ “Tôi yêu Việt Nam”.

* 1. **Ứng dụng bài toán RNN:**

Về cơ bản nếu bạn thấy sequence data hay time-series data và bạn muốn áp dụng deep learning thì bạn nghĩ ngay đến RNN. Dưới đây là một số ứng dụng của RNN:

* **Speech to text**: Chuyển giọng nói sang text.
* **Sentiment classification**: phân loại số sao cho các bình luận, ví dụ: input: “ứng dụng tốt”, output: 4 sao.
* **Machine translation**: Bài toán dịch tự động giữa các ngôn ngữ.
* **Video recognition**: Nhận diện hành động trong video.
* **Heart attack**: Dự đoán đột quỵ tim.

**CHƯƠNG 2**

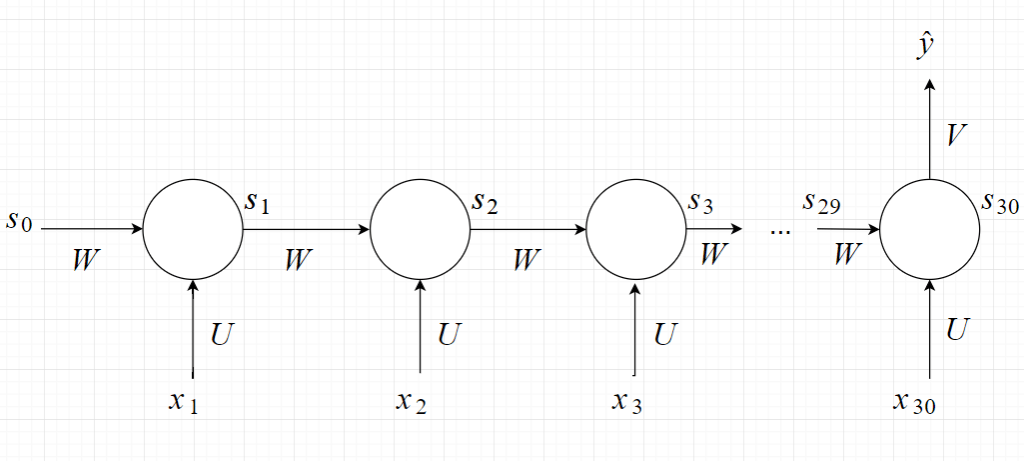
## Mô hình bài toán RNN

**2.1. Mô hình RNN:**

- Bài toán: Nhận diện hành động trong video 30s. Đây là dạng bài toán many to one trong RNN, tức nhiều input và 1 output.

- Input ta sẽ tách video thành 30 ảnh ở mỗi giây. Các ảnh sẽ được cho qua model CNN để lấy ra các feature ([**feature extraction**](https://nttuan8.com/bai-9-transfer-learning-va-data-augmentation/#Feature_extractor)) thành các vector có kích thước n\*1. Vector tương ứng với ảnh ở giây thứ i là ​.

Output là vector có kích thước d\*1, softmax function được sử dụng như trong [**bài phân loại ảnh**](https://nttuan8.com/bai-7-gioi-thieu-keras-va-bai-toan-phan-loai-anh/#Xay_dung_model)**.**

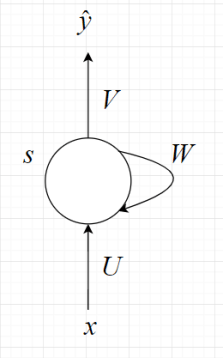


Mô hình RNN cho bài toán

Ta có:

* Mô hình có 30 input và 1 output, các input được cho vào model đúng với thứ tự ảnh trong video *x*1,*x*2,...*x*30​.
* Mỗi hình tròn được gọi là 1 state, state t có input là*xt* và *st*−1 (output của state trước); output là *st*=*f*(*U\*xt*+*W\*st*−1). f là activation function thường là tanh hoặc [ReLU](https://nttuan8.com/bai-10-cac-ky-thuat-co-ban-trong-deep-learning/#Mot_so_activation_thong_dung).
* Có thể thấy st​mang cả thông tin từ state trước (st−1) và input của state hiện tại =>*st* giống như memory nhớ các đặc điểm của các input từ x1 đến xt
* s0 được thêm vào chỉ cho chuẩn công thức nên thường được gán bằng 0 hoặc giá trị ngẫu nhiên. Có thể hiểu là ban đầu chưa có dữ liệu gì để học thì memory rỗng.
* Do ta chỉ có 1 output, nên sẽ được đặt ở state cuối cùng, khi đó s30 học được thông tin từ tất cả các input. . g là activation function, trong bài này là bài toán phân loại nên sẽ dùng softmax.

Ta thấy là ở mỗi state các hệ số W, U là giống nhau nên model có thể được viết lại thành:



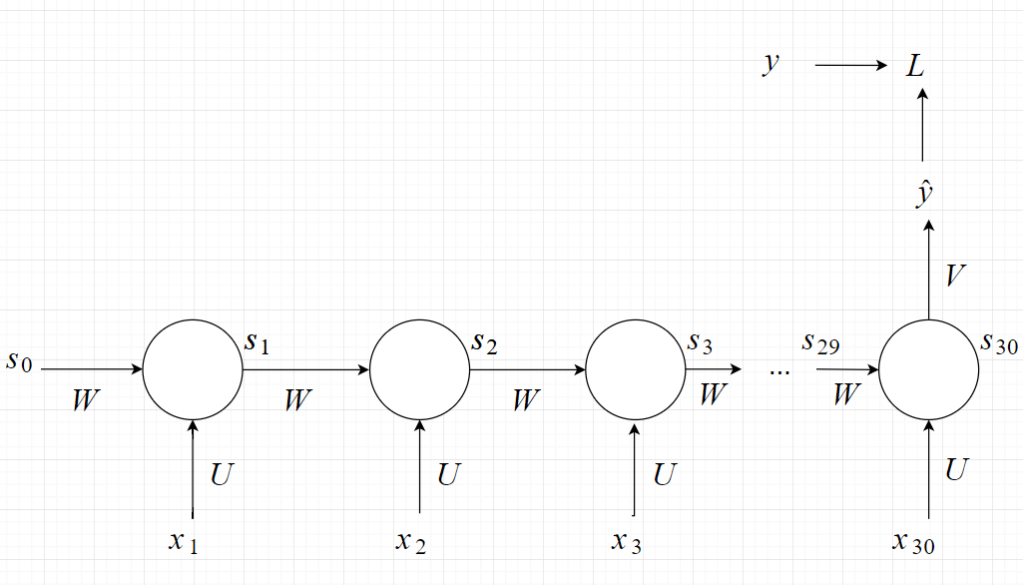
Mô hình RNN rút gọn

Tóm lại:

* xi là vector có kích thước n\*1, si là vector có kích thước m\*1, yi là vector có kích thước d\*1. U là ma trận có kích thước m\*n, W là ma trận có kích thước m\*m và V là ma trận có kích thước d\*m.
* s0 = 0, st=f(U\*xt + W\**s(t*−1)​) với t >= 1
* 

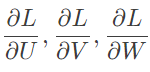
**2.2. Loss function:**

- Loss function của cả mô hình bằng tổng loss của mỗi output, tuy nhiên ở mô hình trên chỉ có 1 output và là bài toán phân loại nên [**categorical cross entropy loss**](https://nttuan8.com/bai-7-gioi-thieu-keras-va-bai-toan-phan-loai-anh/#Loss_function)sẽ được sử dụng.

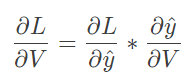


Loss function

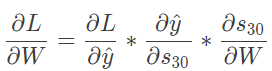
### 2.3. Backpropagation Through Time (BPTT):

Có 3 tham số ta cần phải tìm là W, U, V. Để thực hiện gradient descent, ta cần tính:

Tính đạo hàm với V thì khá đơn giản:



Tuy nhiên với U, W thì lại khác.

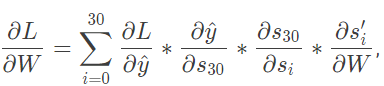
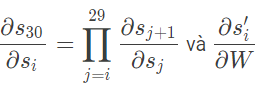


Do s30 = f(W\*s29 + V\*x30) có s29​ phụ thuộc vào W. Nên áp dụng công thức : 

Ta có

, trong đó​​ là đạo hàm của s30​ với W khi coi s29​ là constant với W.

Tương tự trong biểu thức s29 có s28​ phụ thuộc vào W, s28 có s27​ phụ thuộc vào W … nên áp dụng công thức trên và chain rule:

, trong đó và là đạo hàm của si với W khi coi si-1 là constant với W.

- Nhìn vào công thức tính đạo hàm của L với W ở trên ta có thể thấy hiện tượng [vanishing gradient](https://nttuan8.com/bai-10-cac-ky-thuat-co-ban-trong-deep-learning/#Vanishing_va_exploding_gradienthttps://nttuan8.com/bai-10-cac-ky-thuat-co-ban-trong-deep-learning/#Vanishing_va_exploding_gradient) ở các state đầu nên ta cần mô hình tốt hơn để giảm hiện tượng vaninshing gradient => Long short term memory (LSTM) ra đời

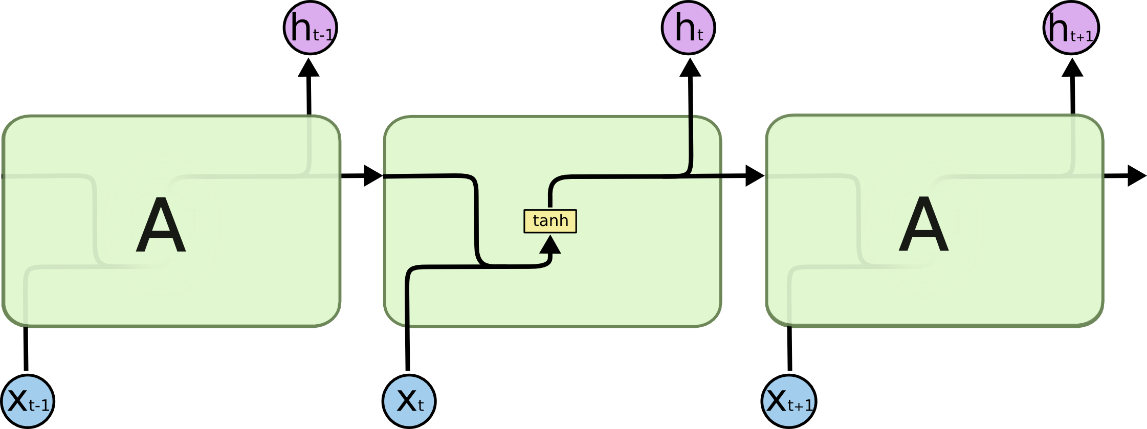
**CHƯƠNG 3**

## LSTM

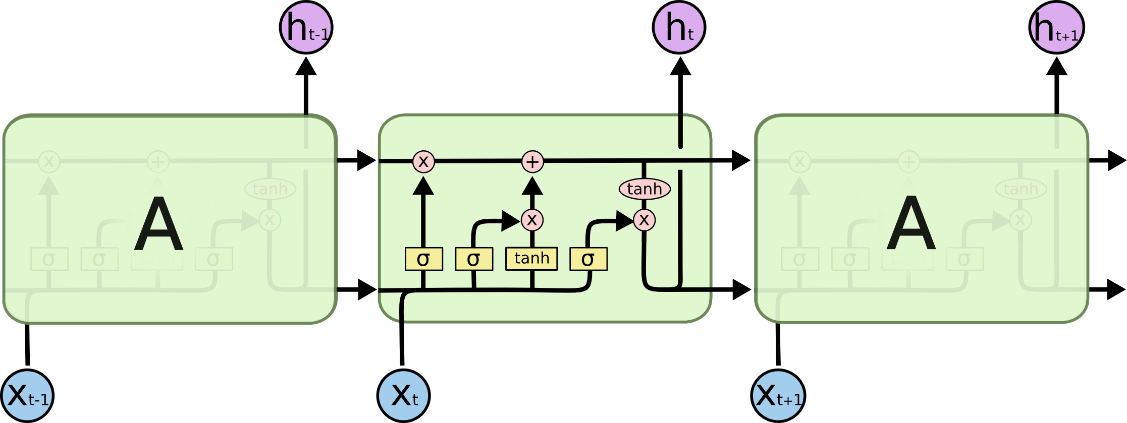
**3.1 Mạng LSTM:**

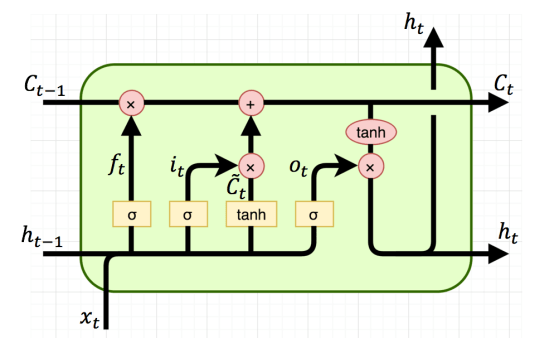
Mạng bộ nhớ dài-ngắn (Long Short Term Memory networks), thường được gọi là LSTM - là một dạng đặc biệt của RNN, nó có khả năng học được các phụ thuộc xa.

Mọi mạng hồi quy đều có dạng là một chuỗi các mô-đun lặp đi lặp lại của mạng nơ-ron. Với mạng RNN chuẩn, các mô-dun này có cấu trúc rất đơn giản, thường là một tầng tanh



LSTM cũng có kiến trúc dạng chuỗi như vậy, nhưng các mô-đun trong nó có cấu trúc khác với mạng RNN chuẩn. Thay vì chỉ có một tầng mạng nơ-ron, chúng có tới 4 tầng tương tác với nhau một cách rất đặc biệt.





* Output: , , ta gọi c là cell state, h là hidden state.
* Input: ,,. Trong đó  là input ở state thứ t của model. , , là output của layer trước. **h** đóng vai trò khá giống như **s**ở RNN, trong khi **c**là điểm mới của LSTM.

Kí hiệu *σ*, tanh ý là bước dùng sigma, tanh activation function. Phép nhân ở đây là [element-wise multiplication](http://nttuan8.com/bai-1:-linear-regression-va-gradient-descent/#Element-wise_multiplication_matrix), phép cộng là cộng ma trận.

tương ứng với **f**orget gate, **i**nput gate và **o**utput gate.

* Forget gate:  =  *σ(*  \* + \* )
* Input gate: *σ(*  \* + \* )
* Output gate: *σ(*  \* + \* )

Nhận xét: 0 < < 1;là các hệ số bias; hệ số W, U giống như trong bài RNN.

, bước này giống hệt như tính   trong RNN.

**,forget gate** quyết định xem cần lấy bao nhiêu từ cell state trước và **input gate** sẽ quyết định lấy bao nhiêu từ input của state và hidden layer của layer trước.

,**output gate** quyết định xem cần lấy bao nhiêu từ cell state để trở thành output của hidden state. Ngoài ra cũng được dùng để tính ra output cho state t.

,​ khá giống với RNN, nên model có **short term memory**. Trong khi đó giống như một băng chuyền ở trên mô hình RNN vậy, thông tin nào cần quan trọng và dùng ở sau sẽ được gửi vào và dùng khi cần => có thể mang thông tin từ đi xa=> **long term memory**. Do đó mô hình LSTM có cả short term memory và long term memory.

**CHƯƠNG 4**

## Màn Hình Kết Quả Code

