BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

SINH TỪ TIẾNG VIỆT VỚI MÔ HÌNH LSTM

Sinh viên: Lê Quang Sang
Mã số: B1606927
Khóa: 42

Cần Thơ, 02/2020

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
BỘ MÔN CÔNG NGHỆ THÔNG TIN

NIÊN LUẬN CƠ SỞ
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

SINH TỪ TIẾNG VIỆT VỚI MÔ HÌNH LSTM

Người hướng dẫn
TS Lâm Nhựt Khang

Sinh viên thực hiện
Lê Quang Sang
Mã số: B1606927
Khóa: K42

Cần Thơ, 02/2020

Lời cảm ơn

Cảm ơn TS. Lâm Nhựt Khang, Bộ môn Công nghệ Thông tin, Khoa Công nghệ Thông tin và Truyền thông, trường Đại học Cần Thơ đã tích cực hướng dẫn, giúp đỡ trong nghiên cứu đề tài này.

Mục lục

Danh mục hình ảnh

Tóm lược
Với sự phát triển của máy tính và công nghệ thông tin hiện nay, máy tính dễ dàng có thể làm những công việc thay cho con người và đặt biệt chúng ta có thể dạy cho máy tính học bằng những gì học được máy tính có thể trả lời những câu hỏi đơn giản đã được dạy hoặc tạo ra một đoạn văn bản với những từ ngữ và câu từ đã được học trước đó. Đây là một mô hình sinh từ tự động tiếng Việt. Chúng ta có thể cung cấp một đoạn văn bản mẫu sau đó mô hình sẽ tự động sinh ra từ ngữ tiếp theo cho đoạn văn bản đó.

Phần giới thiệu

Sinh từ tự động có thể được ứng dụng nhiều trong thực tế như trong các ứng dụng trả lời tự động hoặc đoán ý người dùng để tiếp tục điền đầy vào những từ tiếp theo khi người dùng gõ một vài từ vào máy tính, giúp người dùng dễ dàng thao tác nhanh chóng với những công việc mang tính lập đi lập lại hoặc những câu trả lời tin nhắn đơn giản. Mục tiêu của đề tài này là nghiên cứu về kiến trúc mạng thần kinh hồi quy (RNN) và các biến thể của nó nhằm tạo ra mô hình sinh văn bản dựa trên kiến trúc mạng RNN, cụ thể là mô hình LSTM.

Bố cục của bản báo cáo gồm 3 phần: Phần giới thiệu, Phần nội dung và Phần kết luận, trong đó Phần nội dung gồm có 4 chương:

Chương 1 - Đặc tả yêu cầu: Tìm hiểu mô hình RNN (Mạng nơ-ron hồi quy) các vấn đề của RNN, sự ra đời và cách hoạt động của LSTM

Chương 2 - Thiết kế giải pháp: Trình bài các kiến thức liên quan đến mô hình sinh từ tự động bằng mô hình LSTM

Chương 3 - Cài đặt giải pháp: Cài đặt mô hình sinh từ tự đông dựa trên mô hình LSTM

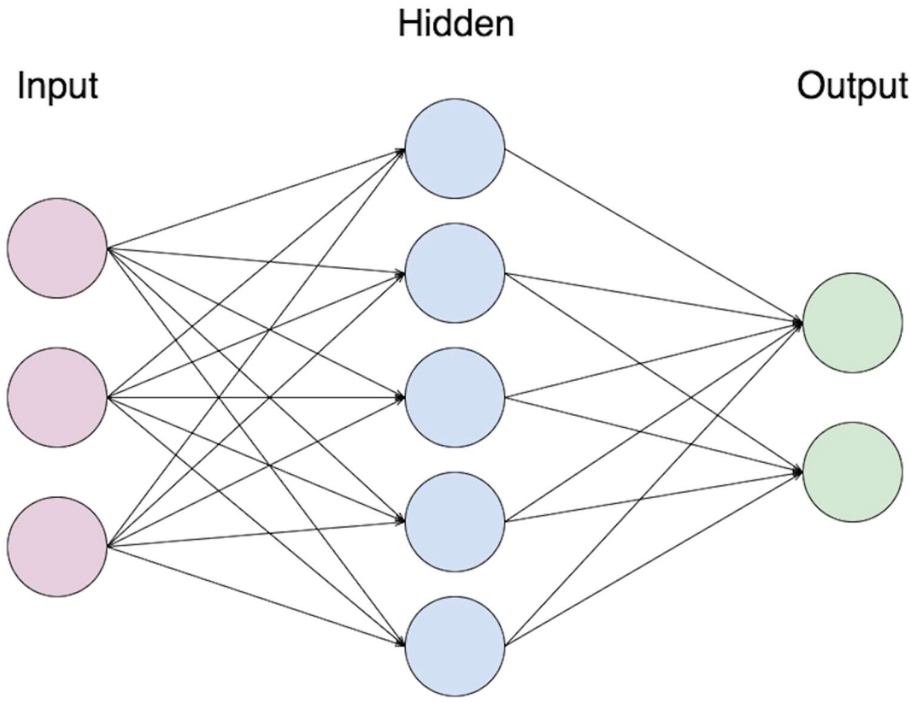
Chương 4 - Đánh giá kiểm thử: Đánh giá mô hình, trình bài mục tiêu và hướng phát triển

Phần nội dung:

Chương 1 - Đặc tả yêu cầu

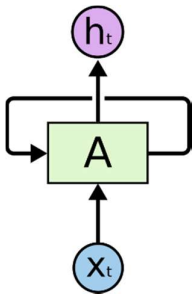
1. Giới thiệu RNN

Để hiểu rõ RNN (Recurrent Neural Networks) là gì chúng ta cùng nhìn lại mô hình mạng nơ-ron truyền thống dưới đây:



Hình 1: Mạng Neron truyền thống

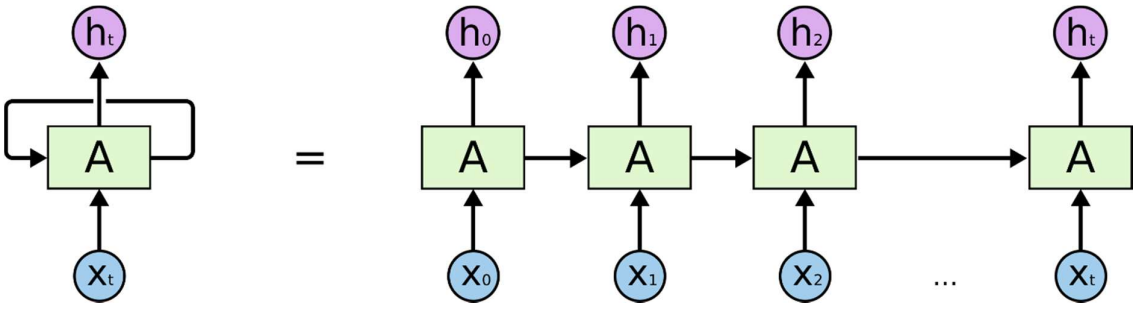
Ta có thể thấy là đầu vào và đầu ra của mạng neuron này là độc lập với nhau. Như vậy mô hình này không phù hợp với những bài toán dạng chuỗi như mô tả, hoàn thành câu,… vì những dự đoán tiếp theo như từ tiếp theo phụ thuộc vào vị trí của nó trong câu và những từ đằng trước nó.



Hình 2: RNN

RNN ra đời với ý tưởng chính là sử dụng một bộ nhớ để lưu lại những thông tin từ những bước tính toán trước đó để đưa ra dự đoán chính xác cho bước hiện tại.

RNN với vòng lặp cho phép thông tin tồn tại bên trong được gọi là các "**cell**". Mô hình trên là một đoạn của mạng, $\boldsymbol{A}$, với input là $\boldsymbol{x}_t$ và ouput là $\boldsymbol{h}_t$. Vòng lặp trong **A** cho phép thông tin đi qua đó và truyền tới bước tiếp theo.
Những vòng lặp này làm cho mạng thần kinh trở nên bí ẩn, nhưng nhìn sâu vào bên trong vấn đề chúng ta có thể thấy mạng RNN có thể được coi nhiều bản sao giống nhau của một mạng. Hãy xem xét khi chúng ta mở vòng lặp của mạng ra.

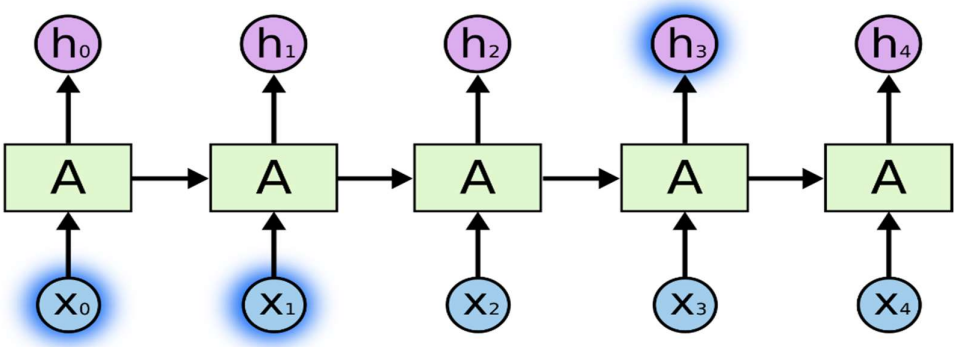


Hình 3: RNN unrolling

Hình trên cho chúng ta thấy các mạng thần kinh tái phát này có liên quan mật thiết đến cái chuỗi tuần tự và danh sách. Và nó được ứng dụng trong các bài toán dữ liệu đầu vào là các chuỗi tuần tự như một câu, một dãy số, các dữ liệu thời gian... Trong một vài năm gần đây mô hình RNN đạt được thành công trong các bài toán như: nhận dạng giọng nói, mô hình ngôn ngữ, mô tả hình ảnh...

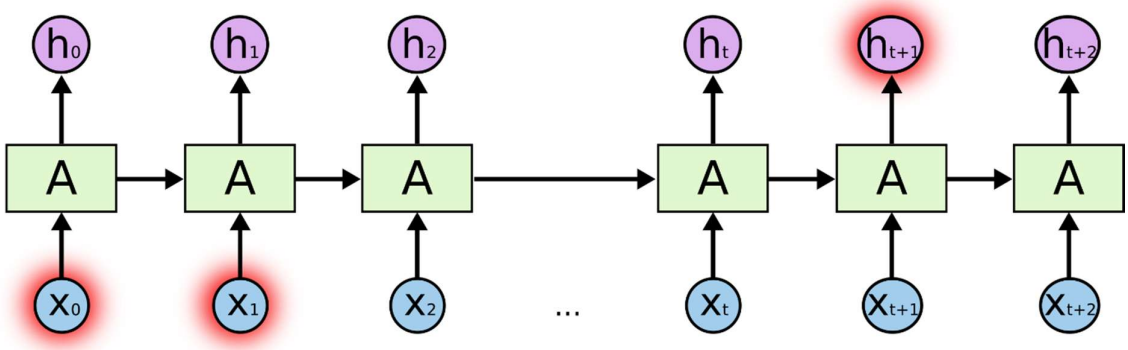
2. *Nhìn nhận về RNN*

Một trong những thế mạnh của RNN là nó có thể liên kế thông tin của một chuỗi tuần tự rất dài trước đó với trạng thái hiện tại, nhưng trong thực tế thì RNN không thể nhớ toàn bộ thông tin của một chuỗi tuần tự rất dài trước đó.
Đôi khi chúng ta chỉ cần một vài thông tin trước đó để biểu diễn cho thông tin hiện tại.
Ví dụ như trong câu "Những đám mây trên *bầu trời*" thì chúng ta không cần bất cứ thông tin liên quan đến ngữ cảnh chúng ta có thể dễ dàng xác nhận từ tiếp theo là "*bầu trời*". Trong trường hợp trên thì khoảng cách giữa các thông tin liên quan là nhỏ, vì thế RNN có thể bằng cách sử dụng thông tin trước đó.



Hình 4: State trong RNN

Nhưng một số trường hợp thì chúng ta cần thêm nhiều thông tin như là ngữ cảnh (context). Ví dụ như trong câu "Tôi lớn lên tại Việt Nam... Tôi nói thông tạo tiếng *Việt*". Trong trường hợp này chúng ta nhận thấy từ có thể là một ngôn ngữ, nhưng để thu hẹp phạm vi biểu diễn thì chúng ta cần một ngữ cảnh "Việt Nam", từ trước đó. Mộ số trường hợp khoảng cách giữa các thông tin trước đó và từ biểu diễn hiện tại có thể là rất xa nhau.



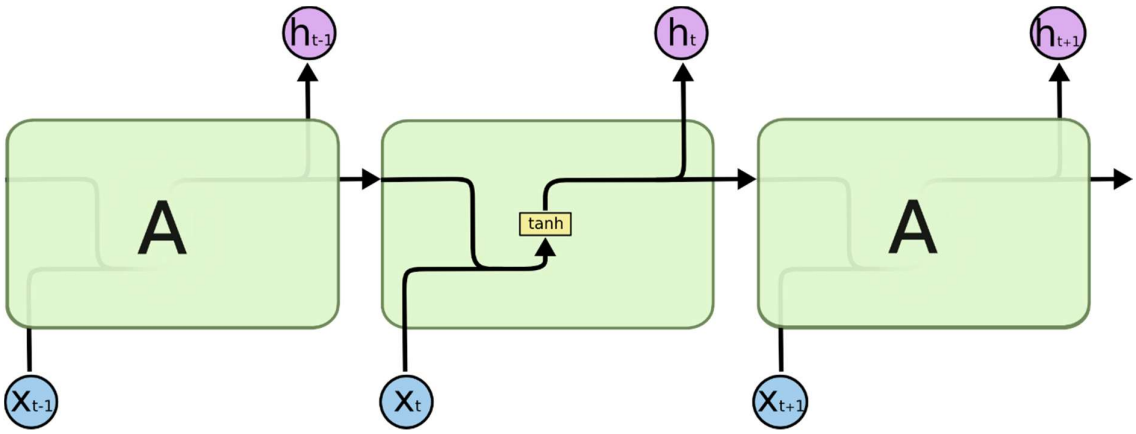
Hình 5: Mất mát thông tin trong RNN

Trên lý thuyết thì RNN có thể kết nối các thông tin phụ thuộc xa nhau như vậy, nhưng thật không may trong thực tế thì RNN không thể làm được điều đó, các vấn đề của RNN được miêu tả cụ thể bởi Hochreiter (1991) [German][1] và Bengio, et al. (1994) [2], họ tìm thấy nhưng lý do cơ bản tại sao RNN lại khó khăn trong vấn đề trên.

3. *Sự ra đời của LSTM*

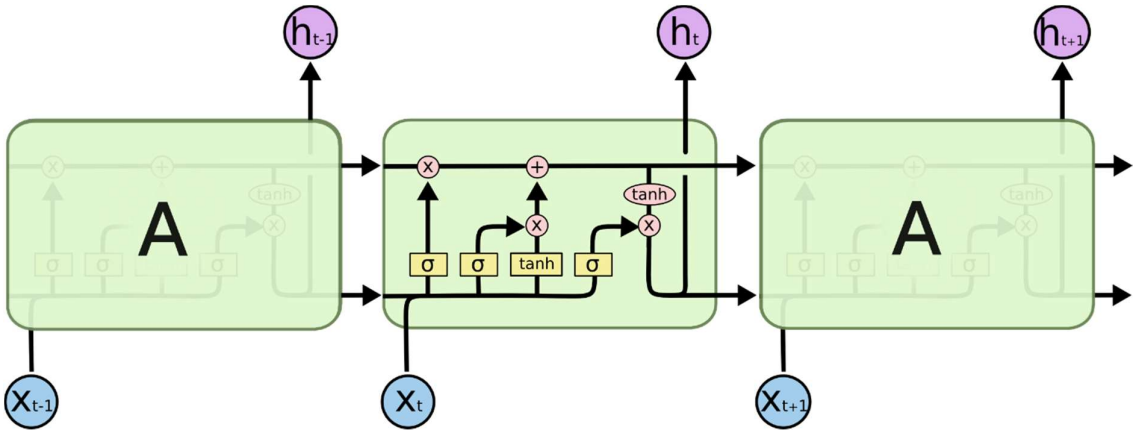
Mạng thần kinh bộ nhớ ngắn hạn - hay thường gọi là LSTMs, về bản chất thì LSTM là một biến thể của mạng RNN truyền thống, với khả năng học được các phụ thuộc ở xa. LSTM được giới thiệu bởi Hochreiter & Schmidhuber (1997) [3], và được nhiều người cải tiến giúp LSTM ngày nay đạt hiệu quả đáng mong đợi.

LSTM được thiết kế để giả quyết vấn đề của RNN truyền thống là phụ thuộc xa. Tất cả các mạng thần kinh tái phát đều có chung một cấu trúc là một chuỗi lập lại của các mạng thần kinh. Ví dụ như cấu trúc của một RNN truyền thống chỉ chứa một ***tanh*** layer.



Hình 6: Kiến trúc mạng RNN

LSTM cũng có cấu trúc chuỗi lặp lại giống như RNN, nhưng mô hình lặp lại có cấu trúc khác nhau, thay vì sử dụng một ***tanh*** layer thì LSTM có tới 4 layer.

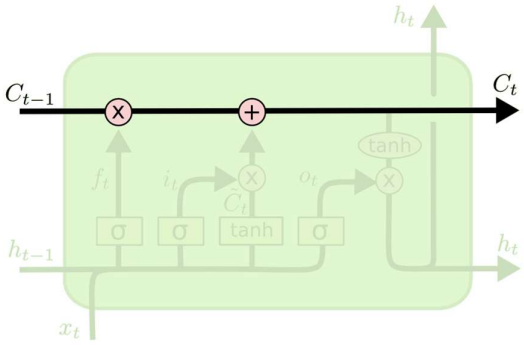


Hình 7: Kiến trúc mạng LSTM

Với các cấu trúc bên trong LSTM là 4 layer như trên hình thì LSTM giúp cho việc ghi nhớ thông tin dễ dàng hơn, cũng như là LSTM cũng có cơ chế cập nhập thêm mới thông tin vào "***cell state***".

4. *Ý tưởng bên trong LSTM*

Ý tưởng chính của LSTMs là "cell state" đường ngang chạy thông suốt trên cùng của mô hình.



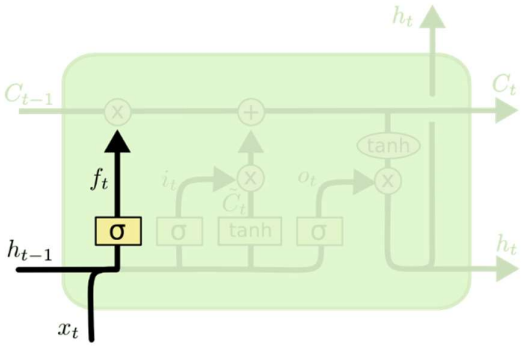
Hình 8: CellState trong LSTM

"cell sate" giống như một băng chuyền nó chạy thông suốt qua toàn bộ chuỗi nó có thể để thông tin truyền đi bên trong nó.

LSTM có khả năng ghi nhớ, loại bỏ thông tin hoặc là thêm mới cập nhật thông tin vào "cell state" một cách cẩn thận với cấu trúc được gọi là "***Gate***"

5. *LSTM hoạt động như thế nào*

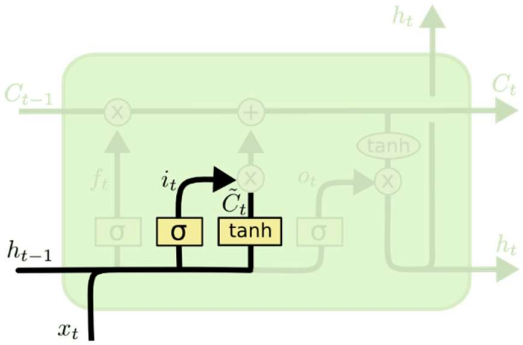
Bước đầu tiên LSTM quyết định thông tin nào mà mạng LSTM loại bỏ từ "cell state". Điều này được quyết định bởi một Gate gọi là "*Forget gate layer*", nó nhìn vào $\mathbf{x}_t$ và $\mathbf{h}_{t-1}$ và đâu ra là một giá trị trong khoảng từ 0 đến 1, 1 thể hiện toàn bộ thông tin được giữ lại, 0 thể hiện loại bỏ toàn bộ thông tin trên $\mathbf{C}_{t-1}$.



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

Hình 9: Input Gate trong LSTM

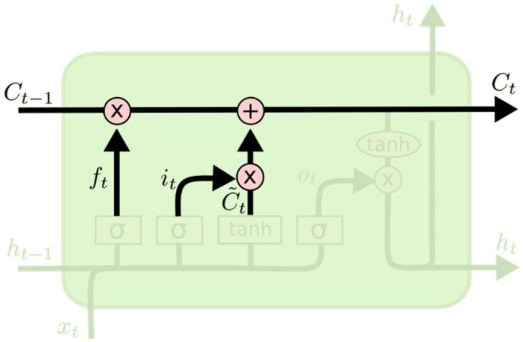
Bước tiếp theo là quyết định thông tin mới nào LSTM lưu giữ trong "cell state". Gồm hai phần, đầu tiên một *sigmoid layer* được gọi là "*input gate layer*" quyết định giá trị nào sẽ được cập nhật. Kế tiếp, một *tanh layer* khởi tạo một vector những giá trị tiềm năng mới, ~$\mathbf{C}_t$, sẽ được thêm vào "cell state", và bước tiếp theo kết hợp hai điều trên lại để tạo một bản cập nhật trang thái.



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

Hình 10: Update gate

Và đây là lúc cập nhật trạng thái "cell state" cũ, $\mathbf{C}_{t-1}$, thành một trạng thái mới $\mathbf{C}_t$. Bước trước đó đã quyết định những gì được cập nhật.
Chúng ta nhân trạng thái cũ $\mathbf{C}_{t-1}$ với $\mathbf{f}_t$, quên đi những gì ta cần quên trước đó, sau đó chúng ta cộng với $\mathbf{i}_t$ * ~$\mathbf{C}_t$. Đây là những giá trị tiềm năng mới được chia theo tỉ lệ quyết định cho việc chúng ta cập nhật mõi giá trị trạng thái như thế nào.

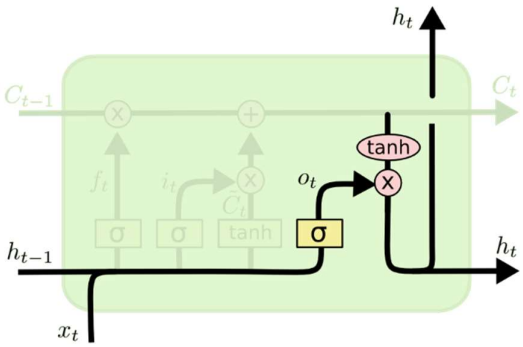


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Hình 11: Update gate 2

Cuối cùng chúng ta quyết định những gì sẽ là đầu ra, đầu ra dựa trên "cell state" của chúng ta, nhưng sẽ là một đầu ra đã được chọn lọc lại. Đầu tiên chúng thực hiện một *sigmoid layer* quyết định phần nào của trạng thái tế bào chúng ta sẽ cho ra. Sau đó

chúng ta đưa trạng thái tế bào (cell state) qua hàm ***tanh*** (các giá trị trong khoảng -1 đến 1) và nhân nó với đầu ra của *sigmoid gate*, do đó chúng ta chỉ cho ra những phần chúng ta đã quyết định.



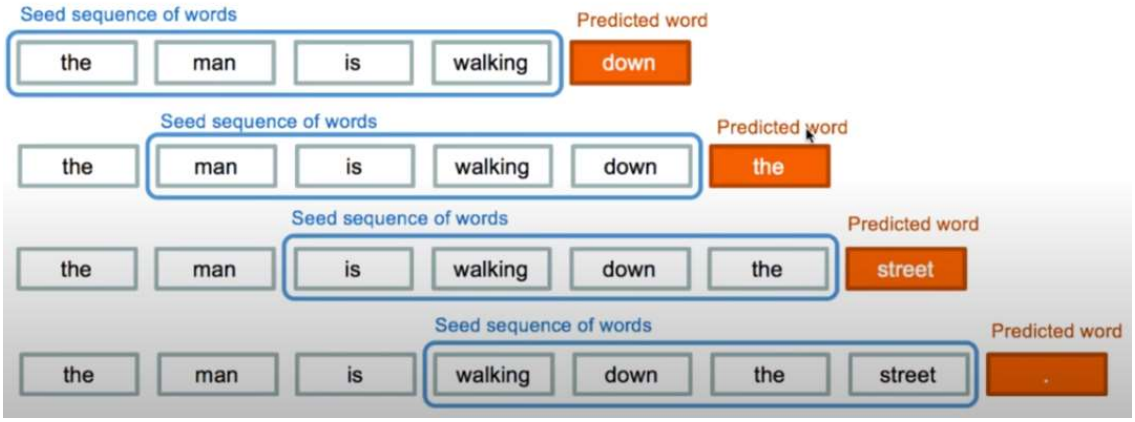
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Hình 12: Output gate

Chương 2 - Thiết kế giải pháp

Mô hình sinh từ tự động bằng mô hình LSTM, nhận dữ liệu đầu vào là các "sequences" có chiều dài 51 từ, với 50 từ đầu tiên làm đầu vào và từ thứ 51 làm nhãn, các nhãn chúng ta đưa về dạng *one-hot-encoding*[4].

Dưới đây là ví dụ cách biểu diễn trên, trong hình bên dưới sử dụng 4 từ làm input là 1 từ làm label. Trong trường hợp của chúng ta thì là 50 thay vì 4 từ như bên dưới.



Hình 13: Cấu trúc input của mô hình

Mô hình LSTM sẽ sử dụng 50 từ đó để dự đoán kết quả của từ thứ 51. Đầu ra của mô hình là một vector phân bố xác xuất của các từ trong bộ từ vựng **y = [vocab_size]**

1. *Tiền xử lý dữ liệu*

Bộ dữ liệu là các bài báo tiếng Việt được thu thập trên internet có dạng:

TPHCM đào tạo đại học cho người sau cai nghiện Đại học Mở-Bán công đã khai giảng lớp đại học từ xa khoa Xã hội và Nhân văn đầu tiên cho học viên, người sau cai nghiện và một số cán bộ, công nhân viên các trung tâm giáo dục dạy nghề của thành phố. Lớp học, được đào tạo theo chế độ tín chỉ, gồm 169 sinh viên, trong đó có 47 học viên sau cai nghiện. Các sinh viên sẽ học tập trung mỗi tuần 2 ngày, thời gian còn lại là tự nghiên cứu.

Nhìn vào dữ liệu trên ta thấy một số vấn đề sau:

- *Chữ viết hoa, thường lẫn lộn*: máy tính phân biệt chữ hoa và chữ thường nên như vậy sẽ tăng độ phức tạp trong quá trình huấn luyện. Do đó chúng ta sẽ chuyển toàn bộ dữ liệu thành chữ thường.
- *Nhiều ký tự thừa*: dấu phẩy, dấu chấm câu, và các dấu câu khác... Do đó chúng ta sẽ loại bỏ các dấu câu trong văn bản.
- *Một số từ viết tắt*: chúng ta sẽ tiến hành chuẩn hóa các chữ viết tắt này.

Sau khi tiền sữ lý chúng ta có kết quả tương tự như sau:

```
['tphcm','đào_tạo','đại_học','cho','người','sau','cai_nghiện','đại_học','mở','bán_công','đã','khai_giảng','lớp','đại_học','từ','xa','khoa','xã_hội','và','nhân_văn','đầu_tiên','cho','học_viên','người','sau','cai_nghiện','và','một_số','cán_bộ','công_nhân_viên','các','trung_tâm','giáo_dục','dạy','nghề','của','thành_phố','lớp_học','được','đào_tạo','theo','chế_độ','tín_chỉ','gồm','169','sinh_viên','trong','đó','có','47','học_viên','sau','cai_nghiện','các','sinh_viên','sẽ','học_tập_trung','mỗi','tuần','2','ngày','thời_gian','còn','lại','là','tự','nghiên_cứu']
```

2. *Xây dựng bộ từ vựng*

Mõi một từ sẽ được gán một số nguyên duy nhất, dựa vào bộ từ điển chúng ta sẽ số hóa các chuỗi huấn luyện thành các chuỗi số để máy tính có thể hiểu và xử lý, chúng ta sẽ xây dựng bộ từ vựng dựa trên những dữ liệu tiền xử lý trước đó của chúng ta. Bằng một hàm trong thư viện *Keras*[5] ta thu được bộ từ vựng như sau:

```
{'của': 1, 'và': 2, 'là': 3, 'có': 4, 'tôi': 5, 'không': 6, 'đã': 7, 'được': 8, 'trong': 9, 'một': 10, 'người': 11, 'cho': 12, 'với': 13, 'những': 14, 'các': 15, 'khi': 16, 'về': 17, 'để': 18, 'anh': 19, 'đến': 20, 'này': 21, 'cũng': 22, 'đó': 23, 'nhưng': 24, 'phải': 25, 'làm': 26, 'nhiều': 27, 'ở': 28, 'ông': 29, 'mình': 30, 'năm': 31, 'ra': 32, 'lại': 33, 'vào': 34, 'thì': 35, 'còn': 36, 'từ': 37, 'bị': 38, 'như': 39, 'phim': 40, 'sẽ': 41, 'tại': 42, 'trên': 43, 'mà': 44, 'theo': 45, 'chỉ': 46, 'rất': 47, 'sau': 48, 'đi': 49, ...}
```

3. *Xây dựng dữ liệu huấn luyện*

Do máy tính làm việc với các con số nên chúng ta phải số hóa chuỗi dữ liệu dạng *text* sang dạng số, sử dụng bộ từ vựng ở trên với mõi từ được đánh một chỉ số duy nhất, chúng ta sẽ số hóa dữ liệu đầu vào thành một chuỗi tuần tự 50 số:

```
['đến','vụ','án','nhắc','đồng','đối_tượng','định_kỳ','đã','bị','tạm','giữ','thông_tin','ban_đầu','cho','biết','lợi_dụng','chính_sách','cho','nợ','thuế_nhập_khẩu','lâm','đã','thành_lập','nhiều','công_ty','dịp','thuê','người_làm','giám_đốc','để','lấy','pháp_nhân','nhập_khẩu','hàng_hóa','xin','ghi','nợ','thuế_nhập_khẩu','diễn','hàng','đi_tiêu_thụ','rồi','cho','đóng_cửa','doanh_nghiệp','hoặc','trốn','khỏi','địa_chỉ','đăng_ký','kinh_doanh']
```

Chuẩn hóa thành số dựa trên bộ từ vựng chúng ta đã tạo trước đó:

```
[98,75,1551,1697,143,306,113,210,541,270,98,506,17,3292,529,2,1062,4467,70,79,7,3362,531,818,27,1378,364,43,1019,114,135,52,4,256,20,75,111,27351,69,229,6713,7,38,353,324,390,569,12,54,878,2254]
```

4. *Xây dựng mô hình*

Mô hình với đầu vào input là một chuỗi 50 từ, mõi từ được biểu diễn bởi một vector trong không gian vector bằng mô hình Word2vec, sau khi đi qua mô hình LSTM sẽ cho đầu ra là một vector với chiều dài bằng số lượng từ vựng trong bộ từ vựng đã xây dựng bên. Vector trên là một vector phân phối xác xuất của các từ trong bộ từ vựng với hàm kích hoạt là "*Softmax*"

Chương 3 – Cài đặt giải pháp

1. *Word2vec*

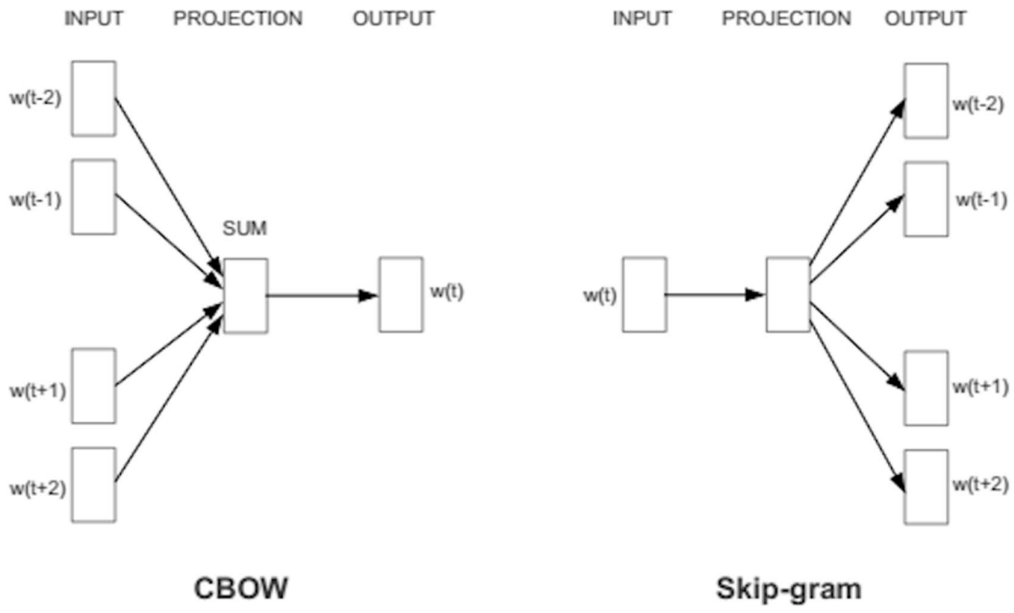
Word2vec là mô hình không gian vector dùng để biểu diễn các từ vựng trong không gian vector, mỗi từ là một vector trong không gian đó. Với số thuộc tính là số chiều được biểu diễn trong không gian vector. Nhằm thể hiện được mối tương đồng, ngữ nghĩa của các từ với nhau, cũng như là mối quan hệ giữa các từ. Những từ có cùng ngữ cảnh sẽ nằm gần nhau trong không gian vector. Trên là những ưu điểm mà theo cách biểu diễn thông thường là "*one-hot*" không thể làm được, và cũng như là sự chiếm dụng dung lượng bộ nhớ khi sử dụng "*one-hot*". Đối với tập dữ liệu quá lớn thì không thể biểu diễn theo "*one-hot*" được. Sau đây là ví dụ biểu diễn từ bằng mô hình word2vec:

		Vua	Hoàng hậu	Phụ nữ	Công chúa
Hoàng gia		0.99	0.99	0.02	0.98
Nam tính		0.99	0.05	0.01	0.02
Nữ tính		0.05	0.93	0.999	0.94
Tuổi		0.7	0.6	0.5	0.1

Hình 14: Biểu diễn từ dưới dạng vector bằng Word2vec

Có hai cách xây dựng mô hình Word2vec:

- Sử dụng ngữ cảnh để dự đoán mục tiêu (CBOW)
- Sử dụng một từ để dự đoán ngữ cảnh (skip-gram)



Hình 15: Các loại mô hình Word2vec

Trong mô hình sinh từ tự động này chúng ta sẽ sử dụng một mô hình Pre-trained Word2vec[6] với kiến trúc Skip-gram[7], và cửa sổ trược ngữ cảnh là 5, và số lượng thuộc tính biểu diễn cho một từ là 400.
Như vậy với đâu vào input là 50 từ, sau khi đi qua mô hình word2vec thì chúng ta có được một ma trận có dạng (50, 400).

2. *Encoder*

Trong mô hình này chúng ta sữ dụng LSTM hai chiều để học được ngữ cảnh trước và sau của một từ tại một "*time step*". Chúng ta sẽ có hai lớp LSTM cho việc mã hóa thông tin. Cả hai lớp LSTM này chúng ta đều trả ra giá trị trạng thái ẩn tại mỗi "*time step"* để cho lớp LSTM tiếp theo sữ dụng các giá trị output này tại mỗi "*time step"*, cũng như là khi đưa vào cơ chế "*Attention*" để tạo sự chú ý đến các từ quan trọng trong chuỗi đầu vào.

3. *Attention*

Kỹ thuật attention (Bahdanau et al., 2016) cho phép bộ giải mã tập trung vào một phần khác nhau từ đầu ra của encoder bằng cách tìm ra chuỗi vector ngữ cảnh của mỗi bước encoder $\mathbf{c}_{1:n}$ với

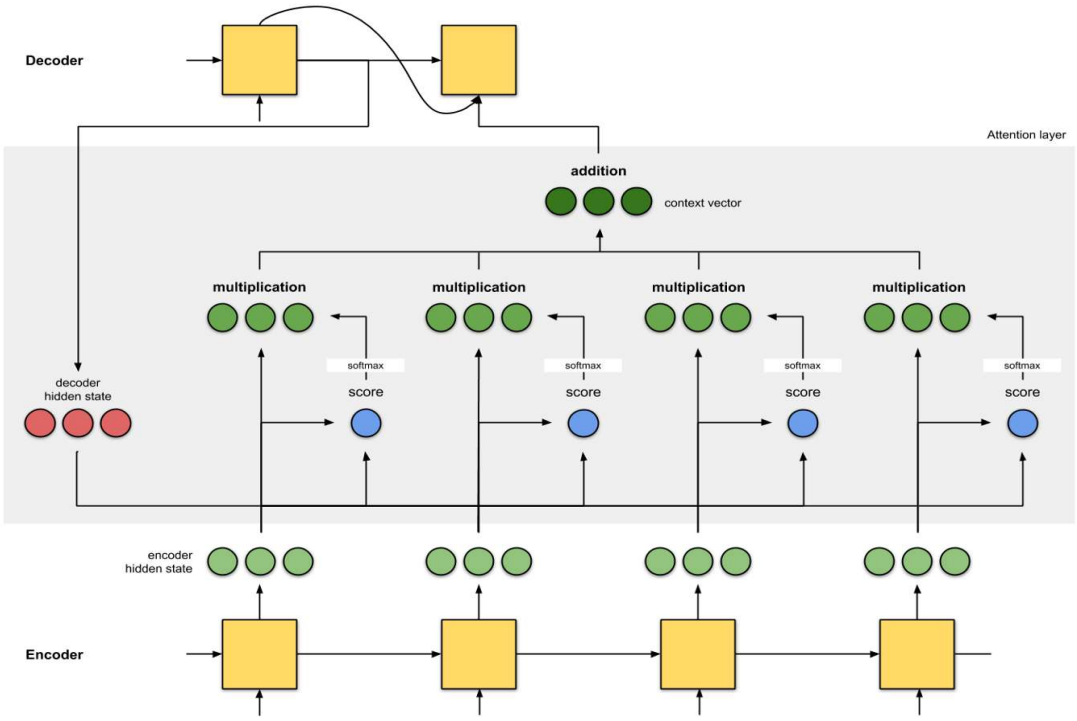
$$c_i = \sum_{j=1}^{n} \alpha_{ij} h_j$$

Trong đó, tập trọng số α_{ij} của mỗi trạng thái ẩn h_j là đầu ra của hàm Sofmax:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=0}^{n} \exp(e_{ik})}$$

$$e_{ij} = q(s_{i-1}, h_j)$$

Với s_{i-1} là trạng thái ẩn tại bước i-1 của decoder



Hình 16: Cơ chế Attention trong LSTM

Chương 4 - Đánh giá kiểm thử

Sử dụng tập dữ liệu các bài báo tiếng Việt bao gồm tất cả các thể loại như: văn hóa, thể thao, chính trị, cuộc sống,... để đánh giá và kiểm tra.

Sau khi thực hiện huấn luyện trên 102,400 sequences, và số lượng từ vựng là 73,788 từ, trainning mức 65/100 epochs, batch_size là 1024, số bước lặp trên mỗi epochs là 100, và thu được các chỉ số loss: 1.4029, accuracy: 0.6335.

1. *Một số kết quả thu được*

```
text = "tình hình vi phạm pháp luật hiện nay diễn biến phức tập trên địa bàn thành phố yêu
ầu người dân chấp hành"
generate_text(model, text, 100)
```

```
'tình_hình vi_phạm pháp_luật hiện_nay diễn_biến phức tập trên địa_bàn thành_phố yêu_cầu
người dân chấp_hành căn_cứ trên nhiều cơ_sở khác bảo vi thuỷ sn 1968 và đặng_hồng_hạnh c
ùng ngụ tại xã 19000 thị_xã 19000 bà_rịa vũng_tàu sống hiện đang được công_an hà_nội thự
c_hiện 5 theo những lời khai của lê_huy_hoàng ma_tuý của hai đối_tượng này đều không đến
mức án của hai bị_cáo khác là nguyễn_thị_bích_thủy và phạm_đức_vinh nhận 1 năm tù về tội
cướp tài_sản hai đồng_phạm 29t3 và phó_chánh cùng bị tand tuyên_phạt 7 năm tù về tội cướ
p tài_sản và tàng_trữ trái_phép vũ_khí quân_dụng thọ là người tổ_chức cung_cấp vũ_khí ch
o đồng_bọn thực_hiện 2 vụ cướp xe dylan trong đêm 15 8'
```

```
text = "chỉ đạo, nhắc nhở, uốn nắn các trường thu chi vô nguyên tắc"
generate_text(model, text, 50)
```

```
'chỉ_đạo nhắc_nhở uốn_nắn các trường thu_chi vô nguyên_tắc đội_ngũ doanh_nghiệp trong đường_dây này đặc_biệt khoản gây tổn_
hại tiền chiếm_lĩnh của các nạn_nhân trong việc điều_tra 1 bị_can này đã ép mượn hàng tuy_nhiên họ đã tạo điều_kiện cho hàn
g trăm hộ dân a2 khu đất trên theo số_liệu của ubnd quận bình_thạnh năm 1995 có đệm việt a2'
```

2. *Nhận xét*

Bản chất mô hình trên đây cơ bản là một mô hình dự đoán từ ("multi-classtification"), Với một chuỗi 50 từ đầu vào dự đoán cho từ thứ 51, và tiếp tục dùng mô hình để tự đoán cho từ tiếp theo 52, và cứ như tiếp tục như thế, mô hình chúng ta có số lượng class dự đoán bằng số lượng từ vựng trong trường hợp này là 73,788 lớp, ví dụ với sau 100 lần dự đoán của mô hình thì ta có một đoạn 100 từ. Do là đầu vào 50 từ nên với số lượng từ đầu vào để dự đoán càng gần 50 thì khả năng mô hình dự đoán từ tiếp theo mang ý nghĩa càng cao.

Phần kết luận

Mô hình trên có thể áp dụng cho các bài toán dự đoán xác suất từ tiếp theo dựa vào những từ trước đó, ví dụ ứng dụng trong các ứng dụng nhập liệu, soạn thảo văn bản, viết lách, viết các review, bình luận,... Giúp cho việc viết bài nhanh hơn và đa dạng về mặt văn phạm.

Mô hình có thể đạt kết quả tốt hơn nếu chúng ta tối ưu hóa bộ từ vựng tốt hơn, giảm số lượng từ vựng, loại bỏ các stopword, các từ trùng lắp, những từ sai chính tả, hoặc lỗi, nhiễu của dữ liệu để đưa số lượng các nhãn dự đoán là thấp nhất có thể thì vector phân phối xác suất đầu ra sẽ chính xác hơn.

Tài liệu tham khảo

[1] http://people.idsia.ch/~juergen/SeppHochreiter1991ThesisAdvisorSchmidhuber.pdf
[2] Y. Bengio ; P. Simard ; P. Frasconi (1994). *Learning long-term dependencies with gradient descent is difficult.*
[3] Hochreiter & Schmidhuber (1997). *Long Short-term Memory*
[4] Goldberg, Yoav; Levy, Omer (2014). "word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method". arXiv:1402.3722 [cs.CL]
[5] Mikolov, Tomas; et al. (2013). "Efficient Estimation of Word Representations in Vector Space"
- Colah's blog. (2015). *Understanding LSTM Networks*
[6] Harris, David and Harris, Sarah (2012-08-07). *Digital design and computer architecture* (2nd ed.). San Francisco, Calif.: Morgan Kaufmann. p. 129. ISBN 978-0-12-394424-5
[7] Keras https://en.wikipedia.org/wiki/One-hot