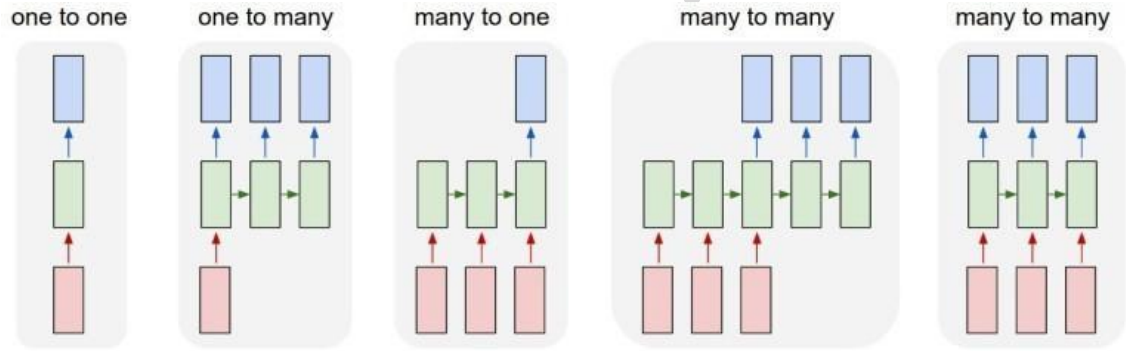
	VIETTEL AI RACE	TD038
	SEQ2SEQ VÀ CƠ CHẾ ATTENTION	Lần ban hành: 1

1. Giới thiệu

Mô hình RNN ra đời để xử lý các dữ liệu dạng chuỗi (sequence) như text, video.



Hình 19.1: Các dạng bài toán RNN

Bài toán RNN được phân làm một số dạng:

- **One to one**: mẫu bài toán cho Neural Network (NN) và Convolutional Neural Network (CNN), 1 input và 1 output, ví dụ với bài toán phân loại ảnh MNIST input là ảnh và output ảnh đấy là số nào.
- **One to many**: bài toán có 1 input nhưng nhiều output, ví dụ với 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, output 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". Để ý là độ dài sequence của input và output có thể khác nhau.

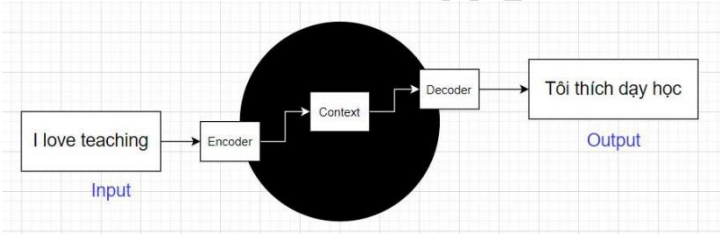
Mô hình sequence to sequence (seq2seq) sinh ra để giải quyết bài toán many to many và rất thành công trong các bài toán: dịch, tóm tắt đoạn văn. Bài này mình sẽ cùng tìm hiểu về mô hình seq2seq với bài toán dịch từ tiếng anh sang tiếng việt.

2. Mô hình seq2seq

Input của mô hình seq2seq là một câu tiếng anh và output là câu dịch tiếng việt tương ứng, độ dài hai câu này có thể khác nhau. Ví dụ: input: I love teaching -> output: Tôi thích dạy học, input 1 câu 3 từ, output 1 câu 4 từ.

Hình 19.2: Seq2seq model

Mô hình seq2seq gồm 2 thành phần là encoder và decoder.

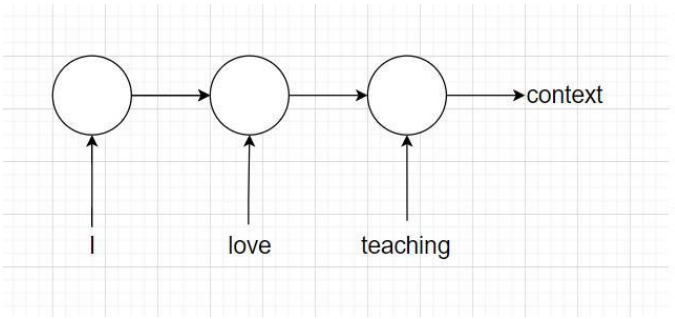


Hình 19.3: Seq2seq model

Encoder nhận input là câu tiếng anh và output ra context vector, còn decoder nhận input là context vector và output ra câu tiếng việt tương ứng. Phần encoder sử dụng mô hình RNN (nói là mô hình RNN nhưng có thể là các mô hình cải tiến như GRU, LSTM) và context vector được dùng là hidden states ở node cuối cùng. Phần decoder cũng là một mô hình RNN với s_0 chính là context vector rồi dần dần sinh ra các từ ở câu dịch.

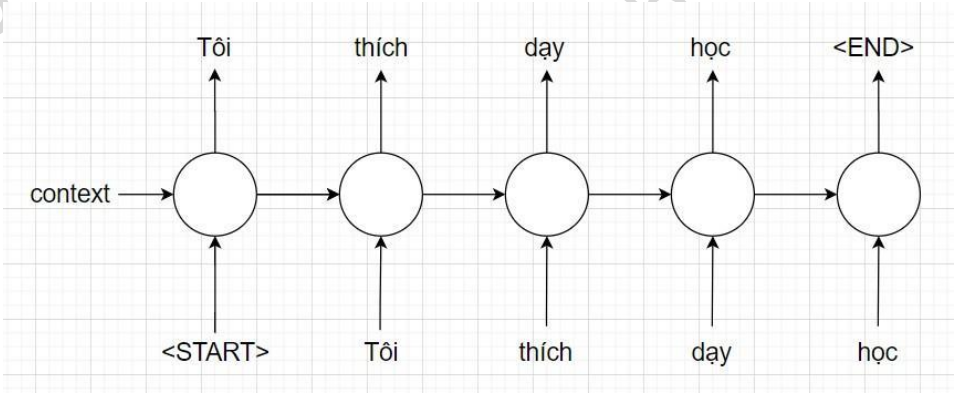
Phần decoder này giống với bài toán image captioning. Ở bài image captioning mình cũng cho ảnh qua pre-trained model để lấy được embedding vector, sau đó cho embedding vector làm s_0 của mô hình RNN rồi sinh ta caption tương ứng với ảnh.

Bài trước mình đã biết model RNN chỉ nhận input dạng vector nên dữ liệu ảnh (từ) sẽ được encode về dạng vector trước khi cho vào model.



Hình 19.4: Mô hình encoder

Các từ trong câu tiếng anh sẽ được embedding thành vector và cho vào mô hình RNN, hidden state ở node cuối cùng sẽ được dùng làm context vector. Về mặt lý thuyết thì context vector sẽ mang đủ thông tin của câu tiếng anh cần dịch và sẽ được làm input cho decoder.



Hình 19.5: Mô hình decoder

2 tag <START> và <END> được thêm vào câu output để chỉ từ bắt đầu và kết thúc của câu dịch. Mô hình decoder nhận input là context vector. Ở node đầu tiên context vector và tag <START> sẽ output ra chữ đầu tiên trong câu dịch, rồi tiếp tục mô hình sinh chữ tiếp theo cho đến khi gặp tag <END> hoặc đến max_length của câu output thì dừng lại.

Vấn đề: Mô hình seq2seq encode cả câu tiếng anh thành 1 context vector, rồi dùng context vector để sinh ra các từ trong câu dịch tương ứng tiếng Việt. Như vậy khi câu dài thì rất khó cho decoder chỉ dùng 1 context vector có thể sinh ra được câu output chuẩn. Thêm vào đó các mô hình RNN đều bị mất ít nhiều thông tin ở các node ở xa nên bản thân context vector cũng khó để học được thông tin ở các từ ở phần đầu của encoder.

=> Cần có cơ chế để lấy được thông tin các từ ở input cho mỗi từ cần dự đoán ở output thay vì chỉ dựa vào context vector => **Attention** ra đời.

3. Cơ chế attention

3.1 Motivation

Attention tiếng anh nghĩa là chú ý, hay tập trung. Khi dịch ở mỗi từ tiếng việt ta cần chú ý đến 1 vài từ tiếng anh ở input, hay nói cách khác là có 1 vài từ ở input có ảnh hưởng lớn hơn để dịch từ đấy.

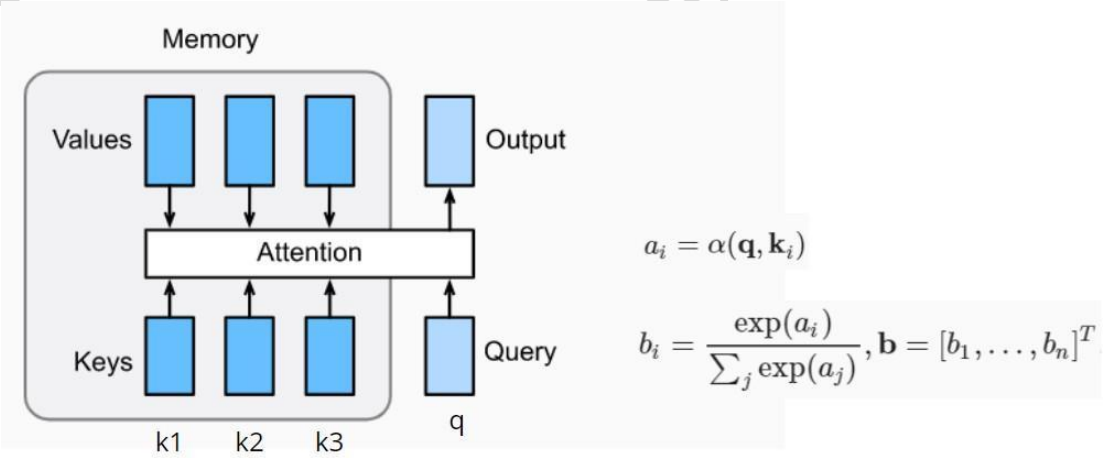
Hình 19.6: Dịch tiếng anh sang tiếng việt, độ quan trọng các từ khi dịch

Ta thấy từ I có trọng số ảnh hưởng lớn tới việc dịch từ tôi, hay từ teaching có ảnh hưởng nhiều tới việc dịch từ dạy và từ học.

=> Do đó khi dịch mỗi từ ta cần chú ý đến các từ ở câu input tiếng anh và đánh trọng số khác nhau cho các từ để dịch chuẩn hơn.

3.2 Cách hoạt động

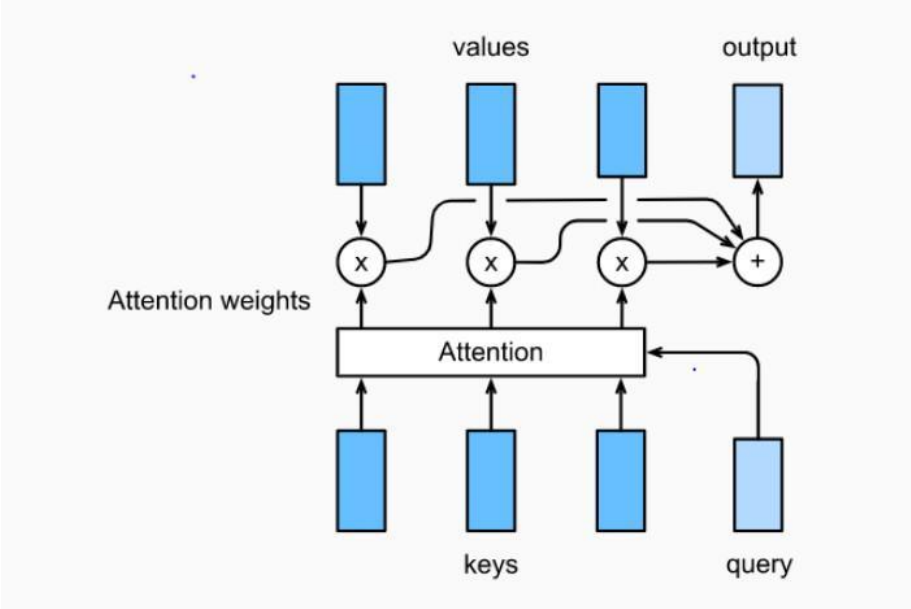
Attention sẽ định nghĩa ra 3 thành phần query, key, value.



Hình 19.7: Các thành phần attention

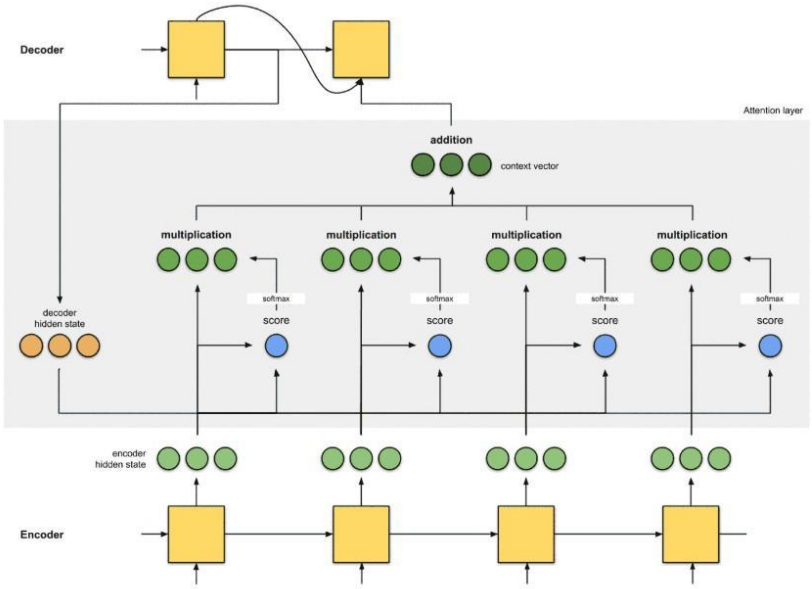
Query (q) lấy thông tin từ từ tiếp theo cần dịch (ví dụ từ dạy). Mỗi từ trong câu input tiếng anh sẽ cho ra 2 thành phần tương ứng là key và value, từ thứ i kí hiệu là k_i, v_i.

Mỗi bộ q, k_i qua hàm α sẽ cho ra a_i tương ứng, a_i chính là độ ảnh hưởng của từ thứ i trong input lên từ cần dự đoán. Sau đó các giá trị a_i được normalize theo hàm softmax được b_i.



Hình 19.8: Các thành phần attention

Cuối cùng các giá trị v_i được tính tổng lại theo hệ số b_i, output = $\sum b_i * v_i$, trong đó N là số từ trong câu input. Việc normalize các giá trị a_i giúp output cùng scale với các giá trị value.



Hình 19.9: Các bước trong attention

Ở phần encoder, thông thường mỗi từ ở input thì hidden state ở mỗi node được lấy làm cả giá trị key và value của từ đấy. Ở phần decoder, ở node 1 gọi input là x_1, output y_1 và hidden state s_1; ở node 2 gọi input là x_2, output y_2. Query là hidden state của node trước của node cần dự đoán từ tiếp theo (s_1). Các bước thực hiện:

- Tính score: $a_i = \alpha(q, k_i)$
- Normalize score: b_i
- Tính output: output_attention = $\sum b_i * v_i$
- Sau đó kết hợp hidden state ở node trước s_1, input node hiện tại x_2 và giá trị output_attention để dự đoán từ tiếp theo y_2.

Name	Alignment score function
Content-base attention	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \text{cosine}[\boldsymbol{s}_t, \boldsymbol{h}_i]$
Additive(*)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \mathbf{v}_a^\top \tanh(\mathbf{W}_a[\boldsymbol{s}_t; \boldsymbol{h}_i])$
Location-Base	$\alpha_{t,i} = \text{softmax}(\mathbf{W}_a \boldsymbol{s}_t)$ Note: This simplifies the softmax alignment to only depend on the target position.
General	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \mathbf{W}_a \boldsymbol{h}_i$ where $\mathbf{W}_a$ is a trainable weight matrix in the attention layer.
Dot-Product	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \boldsymbol{s}_t^\top \boldsymbol{h}_i$
Scaled Dot-Product(^)	$\text{score}(\boldsymbol{s}_t, \boldsymbol{h}_i) = \frac{\boldsymbol{s}_t^\top \boldsymbol{h}_i}{\sqrt{n}}$ Note: very similar to the dot-product attention except for a scaling factor; where n is the dimension of the source hidden state.

Hình 19.10: Một số hàm α hay được sử dụng

Nhận xét: Cơ chế attention không chỉ dùng context vector mà còn sử dụng hidden state ở từng từ trong input với trọng số ảnh hưởng tương ứng, nên việc dự đoán từ tiếp theo sẽ tốt hơn cũng như không sợ tình trạng từ ở xa bị mất thông tin ở context vector.

Ngoài ra các mô hình deep learning hay bị nói là hộp đen (black box) vì mô hình không giải thích được, attention phần nào giúp visualize được kết quả dự đoán, ví dụ từ nào ở output ảnh hưởng nhiều bởi từ nào trong input. Do đó model học được quan hệ giữa các từ trong input và output để đưa ra kết quả dự đoán.

Lúc đầu cơ chế attention được dùng trong bài toán seq2seq, về sau do ý tưởng attention quá hay nên được dùng trong rất nhiều bài toán khác, ví dụ như trong CNN người ta dùng cơ chế attention để xem pixel nào quan trọng đến việc dự đoán, feature map nào quan trọng hơn trong CNN layer,.. Giống như resnet, attention cũng là 1 đột phá trong deep learning. Mọi người để ý thì các mô hình mới hiện tại đều sử dụng cơ chế attention.