Báo cáo VietAI – Final Assignment
Neural Machine Translation

Nguyễn Trung Thành
Tháng 10 năm 2019

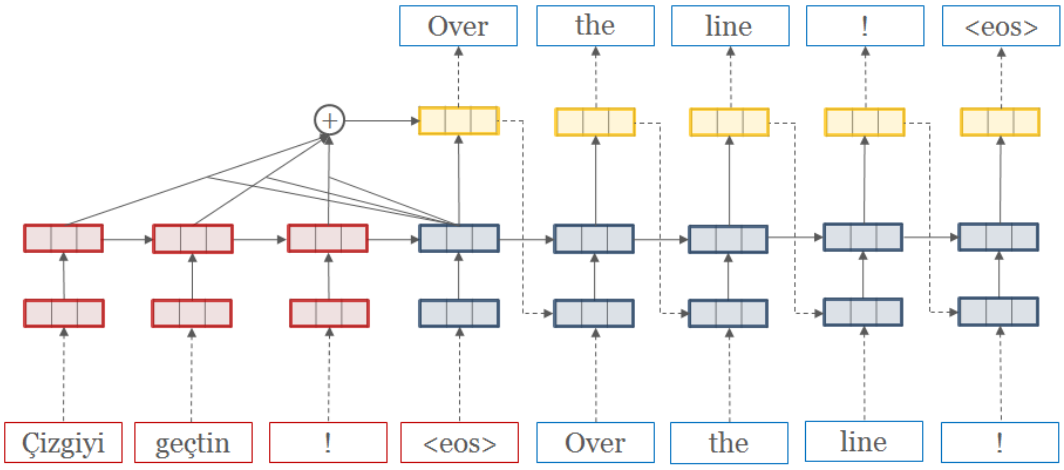
1 Tổng quan về cấu trúc mô hình NMT và phương pháp đánh giá mô hình Machine Translation (Bleu Score)

1.1 Tổng quan về cấu trúc mô hình NMT

Neural Machine Translation (NMT) là phương pháp học end-to-end để dịch tự động, có khả năng khắc phục nhiều điểm yếu của các phương pháp dịch máy truyền thống.

Mỗi ngôn ngữ đều có những đặc trưng riêng về ngữ pháp, cú pháp và ngữ nghĩa vì vậy phương pháp dịch máy truyền thống dựa trên việc dịch từng cụm từ (phrase-by-phrase) còn nhiều hạn chế, làm mất đi sự lưu loát của câu văn. Với NMT là một cách tiếp cận dịch máy (Machine Translation) sử dụng mạng nơ-ron nhân tạo lớn để dự đoán chuỗi từ được dịch, bằng cách mô hình hóa toàn bộ các câu văn trong một mạng nơ-ron nhân tạo duy nhất. Với cách làm này, NMT đã đưa ra những kết quả tốt ngang ngửa với con người.

Cấu trúc của mô hình NMT gồm 2 thành phần chính là: mô hình Sequence-to-Sequence và cơ chế Attention.

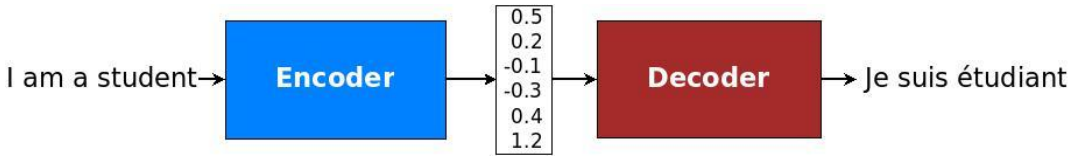


Hình 1: Minh hoạ mô hình Seq2seq kết hợp với cơ chế Attention.

1.1.1 Mô hình Sequence-to-Sequence

Sequence to Sequence Model (Seq2seq) là một mô hình Deep Learning với mục đích tạo ra một output sequence (câu đầu ra) từ một input sequence (câu đầu vào) mà độ dài của 2 sequences này có thể khác nhau. Seq2seq được giới thiệu bởi nhóm nghiên cứu của Google vào năm 2014 trong bài báo Sequence to Sequence with Neural Networks.

Seq2seq gồm 2 phần chính là Encoder (bộ mã hoá) và Decoder (bộ giải mã). Cả hai thành phần này đều được hình thành từ các mạng Neural Networks, trong đó Encoder có nhiệm vụ chuyển đổi input sequence thành một representation với lower dimension còn Decoder có nhiệm vụ tạo ra output sequence từ representation của input sequence được tạo ra ở phần Encoder.

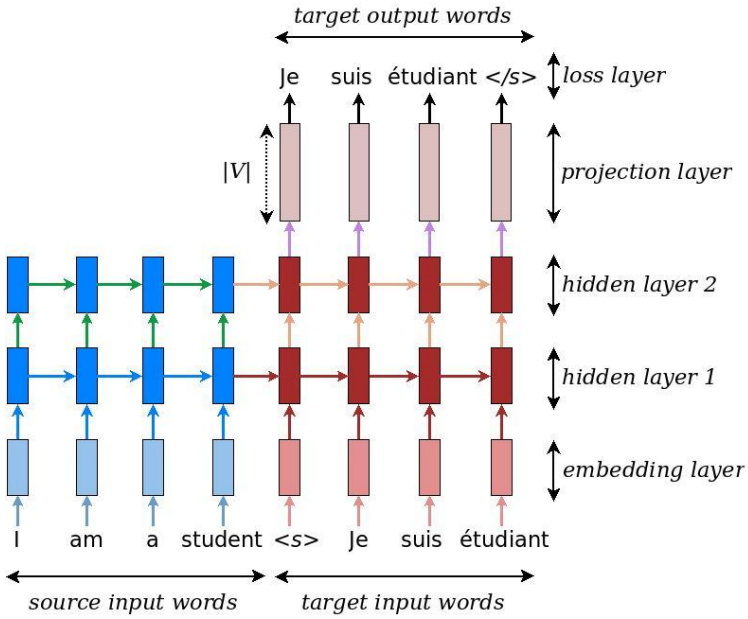


Hình 2: Kiến trúc Encoder-Decoder.

Cụ thể, đầu tiên NMT sẽ đưa input sequence vào Encoder để nén ý nghĩa của câu tạo thành "thought" vector, một chuỗi các số đại diện sẽ chứa ý nghĩa của câu. Sau đó, Decoder xử lý để chuyển vector thành câu được dịch sang ngôn ngữ khác như minh họa trong hình 2.

Tùy từng bài toán cụ thể mà Encoder và Decoder sử dụng các kỹ thuật Deep Learning khác nhau. Ví dụ như trong Machine Translation thì Encoder thường là LSTM, GRU hoặc Bi-directional RNN, còn trong Image Captioning thì Encoder lại là CNN.

Trong báo cáo này, sử dụng Seq2seq cho bài toán Machine Learning. Từ dữ liệu đầu vào là một sequence dưới dạng text, chúng ta sử dụng Embedding Layer để chuyển các từ này sang dạng Word Embedding rồi sử dụng RNN (thường là Bi-directional RNN) để tạo ra một representation của input sequence (trong hình bên dưới là <s>).



Hình 3: Minh hoạ cấu trúc NMT.

Decoder cũng được tạo thành từ RNN và sử dụng output của Encoder làm dữ liệu đầu vào để tạo ra một output sequence. Tuy nhiên khác với Language Modeling, trong Machine Translation chúng ta phải chọn câu văn phù hợp nhất thay vì để RNN cell tạo ra từng từ một. Thông thường

việc lựa chọn output sequence được thực hiện bởi các Search Algorithms với hai phương pháp chính sau:

– Greedy search: chọn từ có xác xuất cao nhất làm output của từng Cell. Ưu điểm của phương pháp này là có tốc độ nhanh, nhưng thường sẽ không tạo ra câu văn hợp lý nhất.
– Beam search: tại mỗi Decoding step, chúng ta chọn n-words (beam width) với xác xuất cao nhất. Ví dụ khi chúng ta chọn beam_width=3 thì tại mỗi Decoding step, ta sẽ giữ lại 3 từ có xác xuất cao nhất rồi lấy từng từ một làm đầu vào cho Decoding step tiếp theo. Cứ như thế lặp lại cho đến khi ta gặp <EOS> đánh dấu việc kết thúc câu.

(Chúng ta có thể coi Greedy search là một trường hợp của Beam search với việc sử dụng beam_width=1).

1.1.2 Vấn đề của mô hình Sequence-to-Sequence

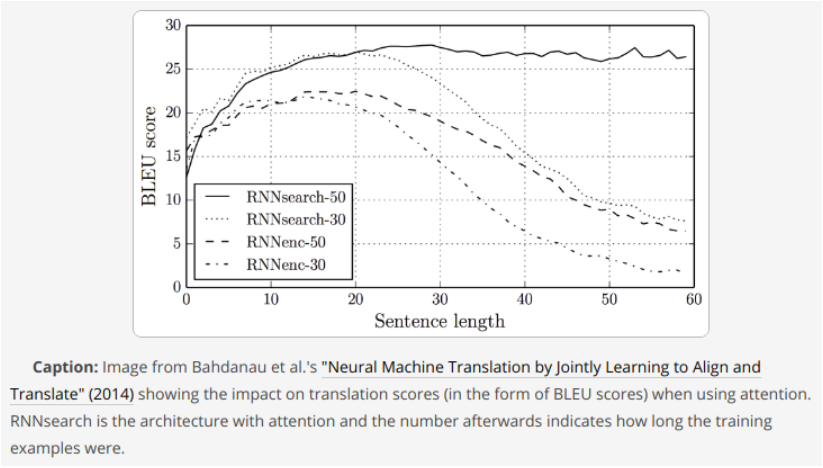
Với Machine Translation thì cả Encoder và Decoder đều được tạo thành từ RNN cell (LSTM hoặc GRU). Về mặt lý thuyết thì LSTM và GRU có thể lưu trữ thông tin của một sequence có độ dài lớn. Tuy nhiên, trong thực tế việc sử dụng một vector representation thường không thể lưu trữ được toàn bộ thông tin của input sequence. Do đó, trong một số bài báo khoa học có trình bày một vài phương pháp giúp tăng độ chính xác cho hệ thống này như:

– Sử dụng Multi-layer với Bi-directional RNN.
– Đảo ngược thứ tự của input sequence. Ví dụ như câu: 'I enjoy eating' thì thứ tự timestep được chuyển thành 'eating enjoy I'.
– Sử dụng input nhiều lần (feeding twice) nhưng vẫn giữ nguyên output.

Tuy nhiên, phương pháp được sử dụng nhiều nhất và làm tăng đáng kể độ chính xác của các hệ thống là sử dụng **Attention Mechanism**.

1.1.3 Cơ chế Attention

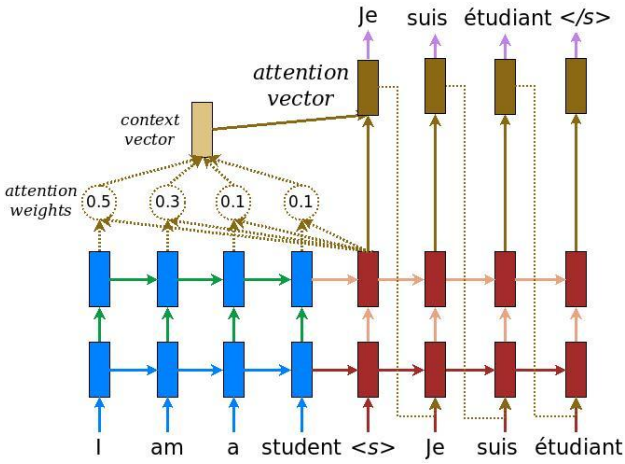
Cơ chế Attention được giới thiệu vào năm 2014 trong bài báo Neural Machine Translation by Jointly Learning to Align and Translate. Hình dưới đây cho thấy kết quả so sánh giữa hệ thống thông thường với hệ thống sử dụng Attention.



Caption: Image from Bahdanau et al.'s "Neural Machine Translation by Jointly Learning to Align and Translate" (2014) showing the impact on translation scores (in the form of BLEU scores) when using attention. RNNsearch is the architecture with attention and the number afterwards indicates how long the training examples were.

Hình 4: Kết quả so sánh giữa hệ thống thường với hệ thống sử dụng Attention.

Nguyên tắc hoạt động chung của Attention Mechanism là tại mỗi Decoding Step, Decoder sẽ chỉ tập chung vào phần liên quan trong input sequence thay vì toàn bộ input sequence. Mức độ tập chung này được thiết lập bởi Attention weights như mô tả trong hình dưới đây:

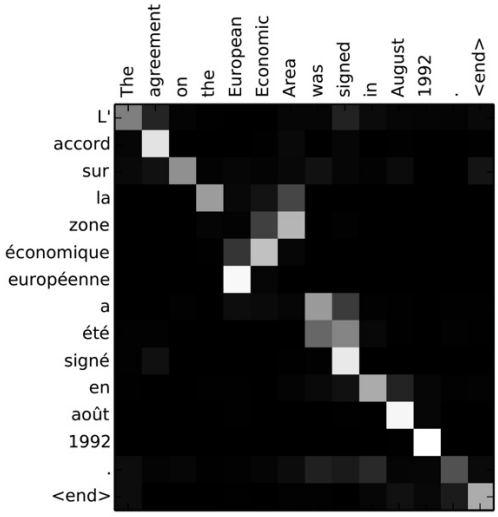


Hình 5: Mô tả cơ chế Attention.

Như vậy, tại mỗi Decoding step, Decoder nhận 3 đầu vào là: Hidden state của decoding step trước, Output của step trước và Attention vector. Attention vector chứa Attention weight của từng từ trong input sequence. Từ nào chứa nhiều thông tin cần thiết cho việc decoding thì sẽ có giá trị weight lớn hơn và tổng các weights của tất cả các từ trong input sequence phải bằng 1.

Giá trị Attention weights được học thông qua quá trình huấn luyện với việc sử dụng input sequence và hidden state của decoding step trước. Mỗi Decoding step có một giá trị Attention vector riêng, do đó với một input sequence có chiều dài 'n' và output sequence có chiều dài 'm', ta phải thực hiện việc tính toán 'n * m' Attention weights. Điều này là có thể chấp nhận được trong các hệ thống Word-based do có số lượng từ không quá nhiều. Tuy nhiên với các hệ thống Character-based thì việc sử dụng Attention sẽ yêu cầu một tài nguyên xử lý lớn.

Hình dưới đây mô tả mối liên hệ giữa Output tại từng Decoding step với các từ trong input sequence. Từ nào có độ sáng càng cao thì có Attention weight càng lớn, cho thấy khi thực hiện Decoding thì hệ thống sẽ chủ yếu tập chung chủ yếu vào các từ này.



Hình 6: Mô tả mối liên hệ giữa Output Sequence và Input Sequence.

1.2 Phương pháp đánh giá mô hình Machine Translation (Bleu Score)

Bilingual Evaluation Understudy Score (BLEU) là một phương pháp dùng để đánh giá chất lượng bản dịch được đề xuất bởi IBM tại hội nghị ACL ở Philadelphie vào tháng 7-2001. Ý tưởng chính của phương pháp là so sánh kết quả bản dịch tự động bằng máy với một bản dịch chuẩn dùng làm bản đối chiếu. Việc so sánh được thực hiện thông qua việc thống kê sự trùng khớp của các từ trong hai bản dịch có tính đến thứ tự của chúng trong câu (phương pháp n-grams theo từ).

Phương pháp này dựa trên hệ số tương quan giữa bản dịch máy và bản dịch chính xác được thực hiện bởi con người để đánh giá chất lượng của một hệ thống dịch. Việc đánh giá được thực hiện trên kết quả thống kê mức độ trùng khớp các n-grams (dãy ký tự gồm n từ hoặc ký tự) từ kho dữ liệu của kết quả dịch và kho các bản dịch tham khảo có chất lượng cao.

Giải thuật của IBM đánh giá chất lượng của hệ thống dịch qua việc trùng khớp của các n-grams đồng thời nó cũng dựa trên cả việc so sánh độ dài của các bản dịch.

Công thức để tính điểm đánh giá của IBM là như sau:

$$score = \exp\left\{\sum_{i=1}^{N} w_i \log(p_i) - \max\left(\frac{L_{ref}}{L_{tra}} - 1{,}0\right)\right\} \qquad (1)$$

- $P_i = \dfrac{\sum_j NR_j}{\sum_j NT_j}$
- NR_j: là số lượng các n-grams trong phân đoạn *j* của bản dịch dùng để tham khảo.
- NT_j: là số lượng các n-grams trng phân đoạn *j* của bản dịch bằng máy.
- $w_i = N^{-1}$
- L_{ref}: là số lượng các từ trong bản dịch tham khảo, độ dài của nó thường là gần bằng độ dài của bản dịch bằng máy.
- L_{tra}: là số lượng các từ trong bản dịch bằng máy.

Giá trị *score* đánh giá mức độ tương ứng giữa hai bản dịch và nó được thực hiện trên từng phân đoạn, ở đây phân đoạn được hiểu là đơn vị tối thiểu trong các bản dịch, thông thường mỗi phân đoạn là một câu hoặc một đoạn. Việc thống kê đồ trùng khớp của các n-grams dựa trên tập hợp các ngrams trên các phân đoạn, trước hết là nó được tính trên từng phân đoạn, sau đó tính lại giá trị này trên tất cả các phân đoạn.

2. Điều chỉnh siêu tham số

- Dữ liệu: **IWSLT English-Vietnamese**.
  https://github.com/tensorflow/nmt#iwslt-english-vietnamese
  Train: 133K examples, vocab=vocab.(vi|en), train=train.(vi|en), dev=tst2012.(vi|en), test=tst2013.(vi|en) - download script.

- Đầu vào cho mô hình được xử lý thông qua **Data Input Pipeline**.
  https://github.com/tensorflow/nmt#data-input-pipeline

- Thực hiện Train trên Google Colab với GPU.

2.1 Các siêu tham số

- attention: sử dụng cơ chế attention (scaled_luong) hoặc không.
- learning_rate: tốc độ học.
- dropout: sử dụng để tránh overfitting.
- optimizer: thuật toán tối ưu hoá (sgd / adam).
- nums_layer: số lớp (độ sâu của mạng).
- nums_unit: số đơn vị (đặc trưng cho kích thước của mạng RNN).
- encoder_type: chiều của encoder – 1 chiều / 2 chiều.
- warmup_scheme – t2t (cho phép khởi tạo với learning_rate nhỏ hơn 100 lần và tăng dần).
- decay_scheme: cho phép khởi tạo learning_rate giảm dần khi về cuối. Một số cơ chế tích hợp sẵn:
  + luong_234: sau 2/3 bước huấn luyện, bắt đầu giảm LR 4 lần, mỗi lần 50%.
  + luong_10: sau 1/2 bước huấn luyện, bắt đầu giảm LR 10 lần, mỗi lần 50% .
  + luong_5: sau ½ bước huấn luyện, bắt đầu giảm LR 5 lần, mỗi lần 50%.
- infer_mode:
  + beam_search: kỹ thuật beam search đã được trình bày ở phần 1.1.1

2.2 Thực hiện thử nghiệm

- Các tham số theo mặc định:

```
--src=vi --tgt=en
--num_train_steps=12000
--steps_per_stats=100
--num_layers=2
--num_units=128
--dropout=0.2
--metrics=bleu
```

- Hiệu chỉnh một số tham số cơ bản:

```
--learning_rate (0.001 / 1.0)
--optimizer (sgd / adam)
--attention (none / scaled_luong)
```

2.3 Kết quả thực hiện

Models	Training Time	Learning Rate	BLEU Score (Test Set)	Attention
SGD	53’	0.001	0.0	No
SGD	42’	1.0	5.3	No
ADAM	44’	0.001	4.2	No
ADAM	overflow, stop early	1.0		No
SGD	1h 6’	0.001	0.0	Yes
SGD	59’	1.0	17.9	Yes
ADAM	55’	0.001	16.9	Yes
ADAM	overflow, stop early	1.0		Yes

Bảng 1: Kết quả Train Model Tiếng Việt sang Tiếng Anh.

Với việc hiệu chỉnh 3 tham số bên trên, kết quả vẫn khá thấp không như kỳ vọng. Nhưng ta có thể thấy được một vài nhận xét sau:

- SGD với Learning Rate = 1.0 tối ưu hơn rất nhiều so với Learning Rate = 0.001.
- ADAM với Learning Rate = 0.001 lại cho kết quả tốt hơn so với Learning Rate = 1.0.
- Với cơ chế Attention cho kết quả tốt hơn khi không sử dụng.
- Thời gian Train với cơ chế Attention tốn nhiều thời gian hơn khi không sử dụng, nhưng thời gian chênh lệnh không quá lớn.

Từ một vài nhận xét trên, ta tiếp tục thay đổi một số tham số khác để kết quả tối ưu hơn:

```
--infer_mode=beam_search
--beam_width=10
--num_units=512
--decay_scheme=luong234
--encoder_type=bi
```

Models	Training Time	Learning Rate	BLEU Score (Test Set)	Attention
SGD	2h10’	1.0	25.1	Yes
ADAM	2h	0.001	22.6	Yes

Bảng 2: Kết quả Train Model Tiếng Việt sang Tiếng Anh sử dụng Beam Search.

**Nhận xét:*

- Từ bảng 1 và bảng 2, trong trường hợp Train Model Tiếng Việt sang Tiếng Anh cho kết quả BLEU Score khi sử dụng SGD với LR = 1.0 cao hơn Model sử dụng ADAM với LR = 0.001.
- Model càng lớn thì thời gian Train càng lâu.

Các kết quả bên trên thực hiện Train dịch từ *Tiếng Việt sang Tiếng Anh*, bây giờ chúng ta sẽ thay đổi để Train dịch từ *Tiếng Anh sang Tiếng Việt*:

```
    --src=en --tgt=vi
```

Models	Training Time	Learning Rate	BLEU Score (Test Set)	Attention
SGD	1h51’	1.0	26.0	Yes
ADAM	1h46’	0.001	25.4	Yes

Bảng 3: Kết quả Train Model Tiếng Anh sang Tiếng Việt sử dụng Beam Search.

**Nhận xét:*

- Kết quả train dịch từ Tiếng Anh sang Tiếng Việt có BLEU Score cao hơn khi dịch từ tiếng Việt sang tiếng Anh.
- Thời gian train khi dịch từ Anh sang Việt nhanh hơn so với dịch từ Việt sang Anh.

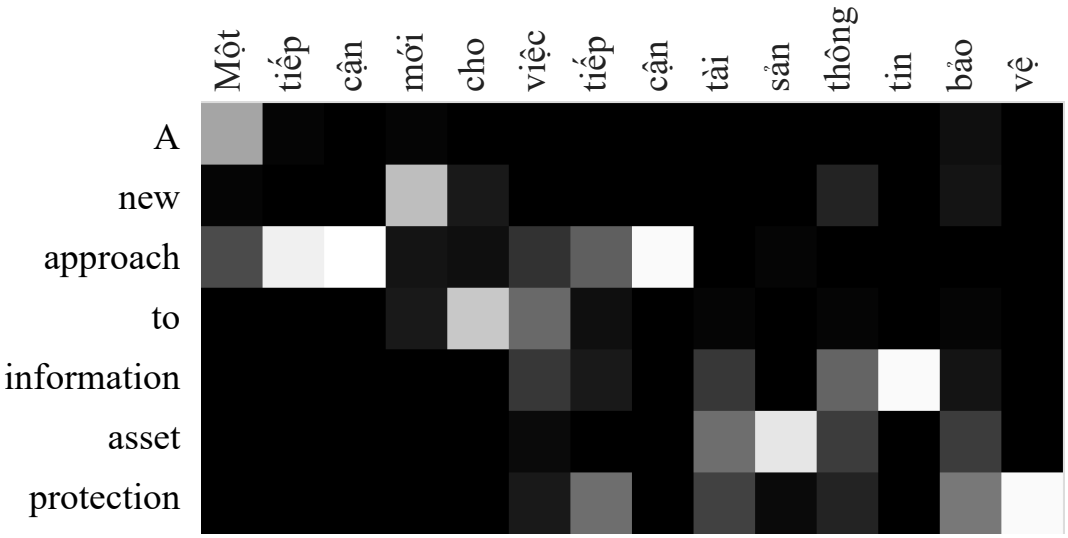
3. Biểu diễn ma trận Attention cho cặp câu Anh – Việt.

Ma trận Attention 1 dưới đây được sinh ra bởi Model SGD được sử dụng trong bảng 3 (dịch từ tiếng Anh sang tiếng Việt) với BLEU Score = 26.0.

- Ma trận Attention 1:

Câu đầu vào: "A new approach to information asset protection".
Đầu ra kỳ vọng: "Một phương pháp mới đối với việc bảo vệ tài sản thông tin".

**Kết quả:* "Một tiếp cận mới cho việc tiếp cận tài sản thông tin bảo vệ".



Ma Trận Attention 1

**Nhận xét:*
- Từ ma trận Attention 1 cho thấy:
  + Từ '*tiếp cận*' tương ứng với từ '*approach*'.
  + Từ 'thông tin' tương ứng với từ 'information'.
  + '*information asset*' có sự liên hệ với '*tài sản thông tin*'.
- Model này dịch chưa sát nghĩa, từ '*bảo vệ*' theo kỳ vọng không nên ở cuối cùng.
- Từ '*tiếp cận*' bị lặp lại.

Từ những nhận xét trên thấy rằng Model này chưa tối ưu mặc dù điểm BLEU Score = 26.0 có giá trị khá cao. Thực hiện Train lại Model SGD với:

```
    --num_layers=4
```

Models	Training Time	Learning Rate	BLEU Score (Test Set)	Attention
SGD	2h45'	1.0	25.5	Yes

Bảng 4: Kết quả Train Model Tiếng Anh sang Tiếng Việt với num_layers = 4.

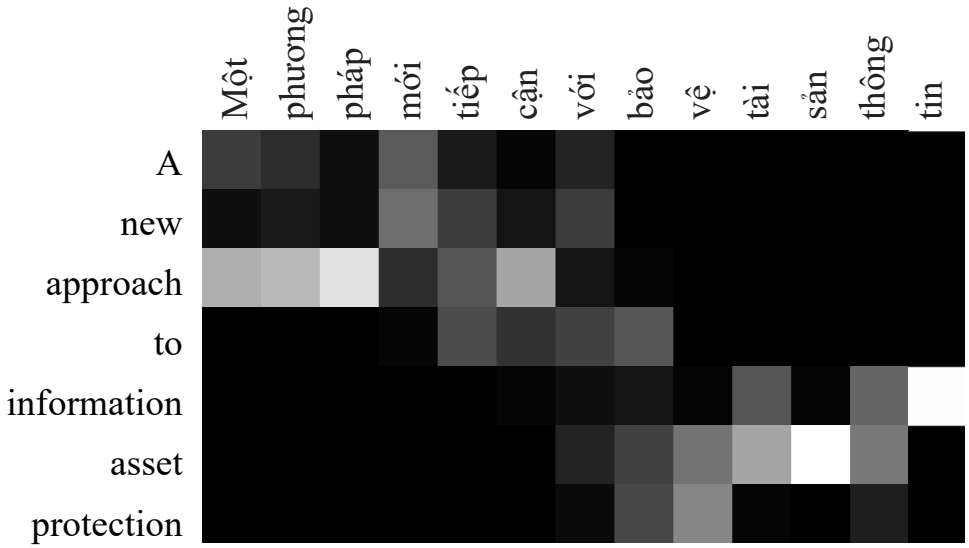
Chú ý: Các ma trận Attention sau được sinh ra bởi Model SGD từ bảng 4 với *num_layers = 4*.

- Ma trận Attention 2:

Thực hiện dịch lại câu trong phần ma trận Attention 1.

Câu đầu vào: “A new approach to information asset protection”.
Đầu ra kỳ vọng: “Một phương pháp mới đối với việc bảo vệ tài sản thông tin”.

**Kết quả:* “Một phương pháp mới tiếp cận với bảo vệ tài sản thông tin”.



Ma trận Attention 2

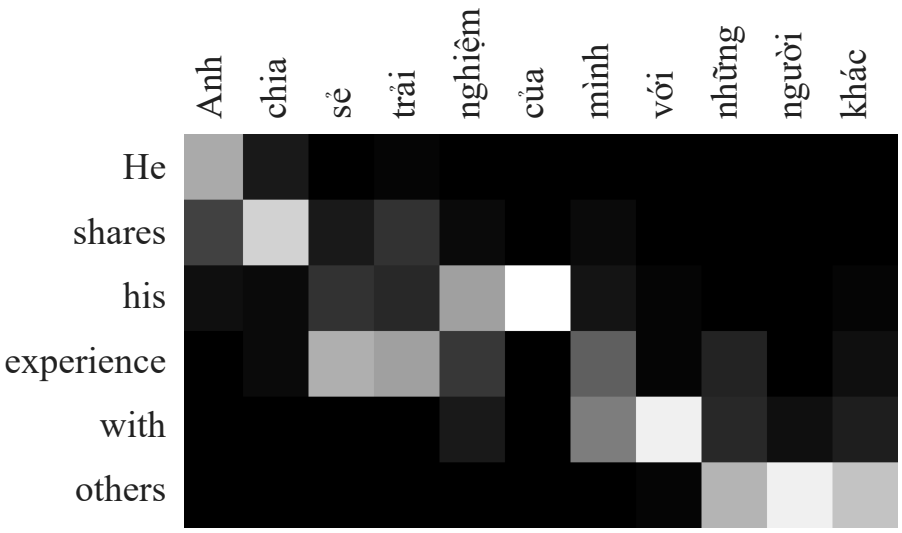
**Nhận xét:*

- Từ ma trận Attention 2 thấy được mối liên hệ rõ giữa 2 phần:
  + ‘*Một phương pháp mới tiếp cận*’ với ‘*A new approach to*’
  + ‘*bảo vệ tài sản thông tin*’ với ‘*asset protection information*’
- Model SGD dùng trong bảng 4 này tốt dịch tốt hơn Model dùng trong bảng 3.

- Ma trận Attention 3:

Câu đầu vào: “He shares his experience with others”.
Đầu ra kỳ vọng: “Anh ấy chia sẻ trải nghiệm với những người khác”.

*Kết quả: “Anh chia sẻ trải nghiệm của mình với những người khác”.



Ma trận Attention 3

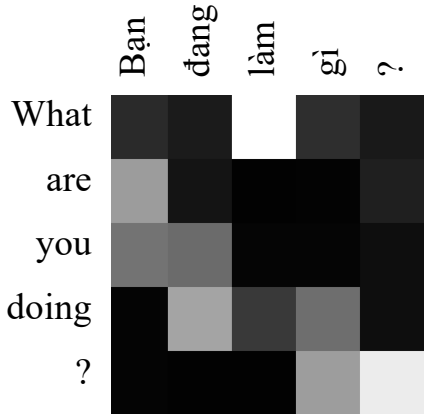
**Nhận xét:*

- Từ '*his*' và '*experience*' dịch sang tiếng Việt có nghĩa '*trải nghiệm của mình'* có mối liên hệ với nhau nên trong ma trận nó có liên hệ với nhau.
- '*with others*' số nhiều dịch sang tiếng Việt là '*với những người khác*'.
- Model này dịch khá sát nghĩa.

- Ma trận Attention 4

Thử nghiệm với câu hỏi: 'What are you doing ?'.
Đầu ra kỳ vọng: 'Bạn đang làm gì ?'.

*Kết quả: 'Bạn đang làm gì ?'.



Ma trận Attention 4

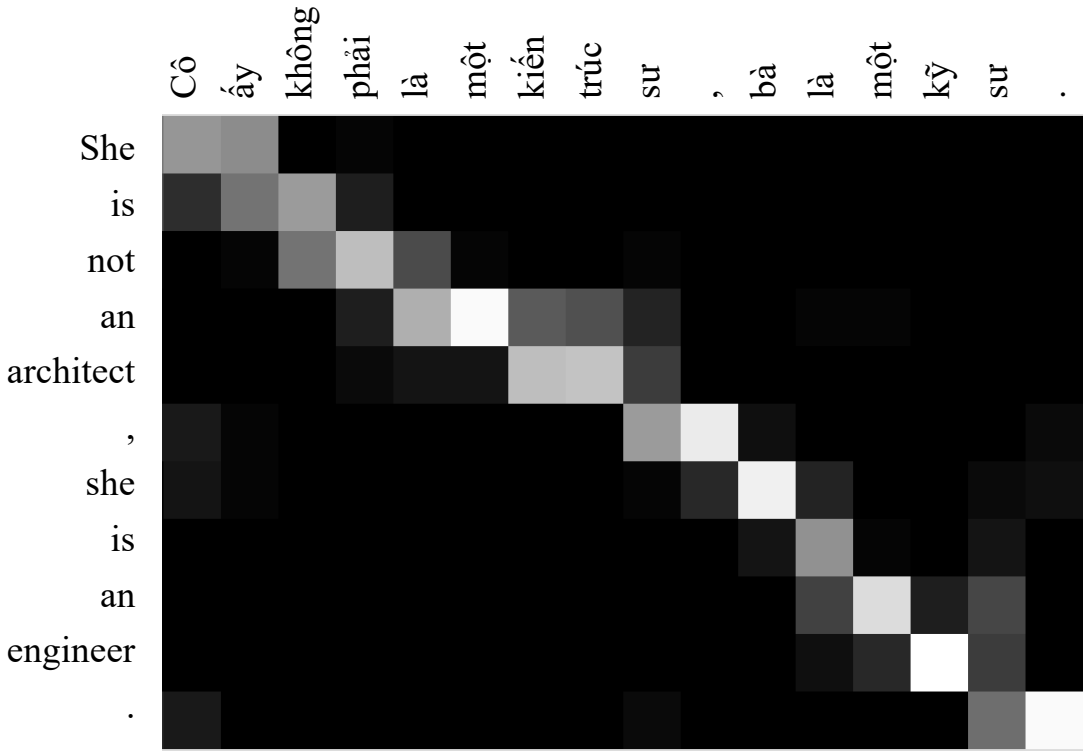
**Nhận xét:*

- Với câu hỏi đơn giản, Model SGD trong bảng 4 dịch rất chính xác.
- Từ ma trận Attention thấy rõ từ '*làm gì*' có liên hệ mạnh với từ '*What*'.

- Ma trận Attention 5

Thử nghiệm với một câu ghép, có một vế phủ định.
Câu đầu vào: “She is not an architect , she is an engineer .”.
Đầu ra kỳ vọng: “Cô ấy không phải là một kiến trúc sư, cô ấy là một kỹ sư”.

*Kết quả: “Cô ấy không phải là một kiến trúc sư , bà là một kỹ sư .”.



Ma trận Attention 5

**Nhận xét:*
- Từ ‘*she*’ thứ 2 có liên kết yếu với từ ‘*cô ấy*’ nên nó đã dịch từ ‘*she*’ này thành ‘*bà*’.
- Nhìn từ ma trận Attention ta thấy các dấu câu có độ sáng hơn các phần dịch, vì cơ bản tiếng Việt và tiếng Anh bộ dấu câu dùng tương đối giống nhau.

4. Tài liệu tham khảo

[1] Neural Machine Translation (seq2seq) Tutorial

[2] Sequence to sequence model - Information Technology Seeker

[3] Attention Mechanism – Information Technology Seeker

[4] Methods and tool for the automatic evaluation of free Online translators