**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**



**TIỂU LUẬN CHUYÊN NGÀNH**

**TÌM HIỂU BÀI TOÁN TẠO CÂU**

**BÌNH LUẬN CHO ẢNH THỜI TRANG**



**Giáo viên hướng dẫn : T.S Nguyễn Thiên Bảo**

**Sinh viên thực hiện : Đỗ Quốc Hùng - 16110097**

**Phạm Thanh Trung - 16110232**

**TP.HCM, tháng 12 năm 2019**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**



**TIỂU LUẬN CHUYÊN NGÀNH**

**TÌM HIỂU BÀI TOÁN TẠO CÂU**

**BÌNH LUẬN CHO ẢNH THỜI TRANG**



**Giáo viên hướng dẫn : T.S Nguyễn Thiên Bảo**

**Sinh viên thực hiện : Đỗ Quốc Hùng - 16110097**

**Phạm Thanh Trung - 16110232**

**TP.HCM, tháng 12 năm 2019**

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP.HCM**  **KHOA CNTT**  **\*\*\*\*\*\*\*** | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh Phúc**  **\*\*\*\*\*\*\*** |

# PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ và tên Sinh viên 1: Đỗ Quốc Hùng MSSV 1: 16110097

Họ và tên Sinh viên 2: Phạm Thanh Trung MSSV 2: 16110232

Ngành: Công nghệ Thông tin

Tên đề tài: Tìm hiểu bài toán tạo câu bình luận cho ảnh thời trang

Họ và tên Giáo viên hướng dẫn: T.S Nguyễn Thiên Bảo

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

- Nhóm đã hoàn thành được các mục tiêu đề ra ban đầu của đề tài, trong khoảng thời gian xác định.

- Nắm đươc kiến thức cơ bản về học sâu như CNN, RNN, cơ chế Attention, kiến trúc NFR.

- Sinh viên nắm được kiến trúc của mô hình tạo câu bình luận cho ảnh thời trang dùng học sâu từ đó áp dụng vào thực tế.

- Sinh viên chạy được demo về tạo câu bình luận cho ảnh thời trang.

2. Ưu điểm:

- Sinh viên chịu khó tìm hiểu, tìm kiếm các tài liệu liên quan đến đề tài

- Sinh viên thực hiện tốt các công việc GVHD giao cho

- Thái độ làm việc: làm việc nghiêm túc, chủ động liên hệ với GVHD

3. Khuyết điểm:

- Chưa có kinh nghiệm về chạy deep learning trên môi trường google colab

- Chưa đánh giá câu bình luận cuối cùng khi chạy thực nghiệm

- Kết quả về mặt trực quan cũng cần cải thiện thêm

- Cần cải thiện mô hình để có bình luận tốt hơn.

4. Đề nghị cho bảo vệ hay không?: Để nghị cho bảo vệ

5. Đánh giá loại:

6. Điểm:

*Tp. Hồ Chí Minh, ngày 07 tháng 12 năm 2019*

Giáo viên hướng dẫn

**T.S Nguyễn Thiên Bảo**

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP.HCM**  **KHOA CNTT**  **\*\*\*\*\*\*\*** | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh Phúc**  **\*\*\*\*\*\*\*** |

# PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên 1: Đỗ Quốc Hùng MSSV 1: 16110097

Họ và tên Sinh viên 2: Phạm Thanh Trung MSSV 2: 16110232

Ngành: Công nghệ Thông tin

Tên đề tài: Tìm hiểu bài toán tạo câu bình luận cho ảnh thời trang

Họ và tên Giáo viên phản biện:

**NHẬN XÉT**

1. Về nội dung đề tài & khối lượng thực hiện :

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

2. Ưu điểm :

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

3. Khuyết điểm :

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

4. Đề nghị cho bảo vệ hay không?

5. Đánh giá loại :

6. Điểm:

*Tp. Hồ Chí Minh, ngày 07 tháng 12 năm 2019*

Giáo viên phản biện

**LỜI CẢM ƠN**

Được sự phân công của quý thầy cô khoa Công nghệ thông tin, Trường Đại Học Sư Phạm Kỹ Thuật TP.HCM, sau gần hết một học kỳ nhóm em đã hoàn thành môn học “Tiểu Luận Chuyên Ngành”.

Để hoàn thành nhiệm vụ được giao, ngoài sự tìm hiểu của bản thân mỗi người trong nhóm còn có sự hướng dẫn tận tình của thầy cô.

Em chân thành cảm ơn thầy Nguyễn Thiên Bảo cùng với cộng tác của thầy là cô Võ Hoàng Anh đã trực tiếp hướng dẫn, hỗ trợ nhóm em trong suốt thời gian thực hiện tiểu luận chuyên ngành. Nhờ những kiến thức, kinh nghiệm mà thầy cô chia sẻ đã giúp chúng em rất nhiều trong việc tìm hiểu và thực hiện tiểu luận chuyên ngành.

Tuy nhiên vì kiến thức chuyên môn còn hạn chế và bản thân còn thiếu nhiều kinh nghiệm thực tiễn nên nội dung của báo cáo không tránh khỏi những thiếu xót, em rất mong nhận sự góp ý, chỉ bảo thêm của quý thầy cô hướng dẫn và thầy cô phản biện để báo cáo này được hoàn thiện hơn.

Một lần nữa nhóm em xin gửi đến thầy cô lời cảm ơn chân thành và chúc thầy cô dồi dào sức khoẻ!

**Sinh viên thực hiện**

Đỗ Quốc Hùng

Phạm Thanh Trung

**LỜI NÓI ĐẦU**

Trong cuộc sống hiện đại ngày nay, thời trang ngày càng được mọi người chú trọng hơn. Việc mặc gì ra đường một phần nào đó thể hiện tính cách cũng như con người của bạn. Nhưng để phối được một bộ quần áo vừa đẹp vừa hợp thời trang thì đó vẫn là một vấn đề khá nhức nhối hiện nay. Để giải quyết vấn đề đó, chúng ta cần một người có kiến thức về thời trang đưa ra những nhận xét cũng như góp ý về trang phục của mình từ đó cải thiện được vấn đề phối đồ hợp thời trang. Áp dụng công nghệ để giải quyết vấn đề nói trên, bài toán tạo câu bình luận cho ảnh thời trang ra đời từ đó.

Với bài toán tạo câu bình luận cho ảnh thời trang, người dùng có thể nhận được bình luận khi phối một quần với một áo bất kỳ. Và để nói rõ hơn, chúng ta sẽ cùng tìm hiểu kỹ hơn về bài toán này cũng như những hạn chế và lợi ích mà nó mang lại .

MỤC LỤC

[Chương I. Giới thiệu](#_Toc532560538) 5

[1. Đặt vấn đề](#_Toc532560539) 5

[2. Mục tiêu](#_Toc532560540) 5

[3. Phạm vi đề tài](#_Toc532560541) 6

[Chương II. Tổng quan về học sâu (Deep Learning)](#_Toc532560545) 7

[1. Giới thiệu về Deep Learning](#_Toc532560546) 7

[2. Mạng tích chập (Convolutional Neural Network)](#_Toc532560547) 7

[2.1. Đặc trưng (Feature)](#_Toc532560575) 8

[2.2. Tích chập (Convolutional)](#_Toc532560575) 9

[2.3. Pooling](#_Toc532560575) 10

[2.4. Fully connected layers](#_Toc532560575) 11

[3. Mạng hồi quy (Recurrent Neural Network)](#_Toc532560572) 12

[3.1. RNN (Recurrent Neural Network)](#_Toc532560575) 12

[3.2. GRU (Gated Recurrent Unit)](#_Toc532560575) 14

[Chương III. Phương pháp đề xuất](#_Toc532560569) 15

[1. NFR framework](#_Toc532560570) 15

[2. Kiến trúc của NFR framework](#_Toc532560571) 15

[3. Encoder – Decoder Architecture](#_Toc532560573) 16

[4. Sequence to Sequence Model](#_Toc532560574) 17

[5. Attention Model](#_Toc532560588) 19

[6. Transformer Architecture](#_Toc532560593) 23

[6.1. Mô hình kiến trúc của Transformer](#_Toc532560575) 23

[6.2. Input Embedding và Positional Embedding](#_Toc532560575) 25

[6.3. Giai đoạn encoder](#_Toc532560575) 26

[6.3.1. Seft-Attention trong Transformer](#_Toc532560575) 27

[6.3.2. Scaled Dot-Product Attention](#_Toc532560575) 31

[6.3.3. Multi head seft attention](#_Toc532560575) 33

[6.3.4. Position-wise Feed Forward Networks](#_Toc532560575) 35

[6.3.5. Residual connection](#_Toc532560575) 35

[6.4. Giai đoạn encoder tiếp theo](#_Toc532560575) 36

[6.5. Ưu điểm và nhược điểm của kiến trúc Transformer](#_Toc532560575) 37

[7. BERT](#_Toc532560594) 38

[Chương IV. Hiện thực hóa hệ thống](#_Toc532560598) 41

[1. Dataset](#_Toc532560599) 41

[2. Môi trường](#_Toc532560600) 41

[3. Kết quả](#_Toc532560601) 42

[Chương V. Tổng kết và hướng phát triển](#_Toc532560607) 43

[1. Các công việc đã làm](#_Toc532560608) 43

[2. Hạn chế](#_Toc532560609) 43

[3. Hướng phát triển](#_Toc532560610) 43

[TÀI LIỆU THAM KHẢO](#_Toc532560615) 44

**DANH MỤC HÌNH VẼ**

[**Hình 1**: Minh họa đơn giản nguyên lý hoặt động của CNNs](#_Toc532560509) 8

[**Hình 2:** Cách so sánh trong CNN](#_Toc532560511) 8

[**Hình 3**: Nguyên lý tạo ra feature map](#_Toc532560512) 9

[**Hình 4**: Thực hiện toàn bộ quá trình tích chập cho từng feature khác](#_Toc532560513) 9

[**Hình 5**: ReLU activation function](#_Toc532560514) 10

[**Hình 6**: Cách hoạt động của Pooling](#_Toc532560515) 10

[**Hình 7**: Thu gọn những feature map quan trọng sau khi qua pooling](#_Toc532560516) 11

[**Hình 8**: Miêu tả quy trinh các lớp của CNN](#_Toc532560517) 11

[**Hình 9**: Kiến trúc của mô hình RNN](#_Toc532560518) 13

[**Hình 10**: Gated Recurrent Unit (GRU)](#_Toc532560519) 14

[**Hình 11**: Sơ đồ kiến trúc tổng quan của neural fashion recommendation (NFR)](#_Toc532560519) 15

[**Hình 12**: Sơ đồ kiến trúc Encoder – Decoder Architecture](#_Toc532560520) 17

[**Hình 13**: Encoder-decoder model dịch câu “she is eating a green apple” sang tiếng Trung](#_Toc532560521) 18

[**Hình 14**: Mô hình đang tạo ra từ yt tại thời điểm t bằng cách áp dụng cơ chế Attention](#_Toc532560522) 20

[**Hình 15**: Mức độ tương quan (correlation) giữa dữ liệu nguồn (source) và dữ liệu dự đoán (target)](#_Toc532560523) 22

[**Hình 16**: Kiến trúc mô hình Transformer](#_Toc532560524) 24

[**Hình 17**: Các khối encoder/decoder được xếp chồng nhau trong mô hình Transformer](#_Toc532560525) 24

[**Hình 18**: Một lớp của giai đoạn encoder](#_Toc532560526) 27

[**Hình 19**: Mã hóa từ "it" trong encoder layer thứ 5](#_Toc532560527) 28

[**Hình 20**: Miêu tả việc tạo ra các vector query, key, value cho một từ bằng việc nhân embedding vector với các matix trong số tương ứng](#_Toc532560528) 29

[**Hình 21**: Miêu tả việc tạo ra ma trận query, key, value cho câu dữ liệu nguồn, mỗi hàng trong ma trận X thể hiện 1 từ trong câu dữ liệu nguồn](#_Toc532560529) 29

[**Hình 22**: Giai đoan Encoder ở layer thứ 1, miêu tả việc xử lý các vector đại diện qua từng sublayer](#_Toc532560530) 30

[**Hình 23**: Scaled Dot-Product Attention](#_Toc532560531) 32

[**Hình 24**: Tính điểm attention score bằng hàm tương thích Scaled Dot-Product Attention, con số trong hình là minh họa](#_Toc532560532) 32

[**Hình 25**: Multi-Head Attention gồm nhiều lớp h attention chạy song song](#_Toc532560533) 34

[**Hình 26**: Miêu tả chi tiết Multi head Attention xử lý dữ liệu song song qua các multi-head để thành các vector đại diện](#_Toc532560534) 34

[**Hình 27**: Miêu tả mối quan hệ của từ “it” với từ còn lại trong câu, sau khi đã sử dụng multi-head attention](#_Toc532560535) 35

[**Hình 28**: Giai đoạn decoder](#_Toc532560536) 36

[**Hình 29**: Sau khi kết thúc giai đoạn encoder, ma trận K và V được đưa vào giai đoạn decoder. Mỗi bước trong giai đoạn decoder sẽ dự đoán ra 1 từ trong chuỗi đầu ra](#_Toc532560536) 37

[**Hình 30**: Biểu diễn đầu vào pre-training của BERT](#_Toc532560536) 39

[**Hình 31**: Ứng dụng Bert vào bài toán phân loại từ](#_Toc532560537) 40

# Chương I. Giới thiệu

1. **Đặt vấn đề**

Thị lực máy tính (Computer vision) là một nhánh trong lĩnh vực trí tuệ nhân tạo (Artificial Intelligence) và Khoa học máy tính (Computer science). Lĩnh vực này giúp máy tính có khả năng thị giác như con người, giúp máy tính có thể nhận diện và hiểu biết một hình ảnh mang tính điện tử.

Xã hội ngày một phát triển, cùng theo đó là nhịp sống của mọi người cũng bận rộn theo. Xu hướng mua hàng online, ngày càng được giới trẻ và những người không có nhiều thời gian mua sắm lựa chọn. Đây cũng chính là lĩnh vực mà rất nhiều tìm năng để khai thác, chính vì vậy để năng cao chất lượng phục vụ cũng như tạo sự quan tâm đến cho khách hàng ngày càng được xem trọng và được đầu tư. Lấy ý tưởng từ việc gợi ý những mẫu thời trang cho người mua hàng trên những trang mua sắm online nổi tiếng như Amazon, Lazada, Alibaba, Tiki,… những hệ thống gợi ý này tập trung vào việc dựa vào lịch sử người dùng đã tìm kiếm từ đó gợi ý ra những sản phẩm tương tự cho người dùng. Để mở rộng chức năng gợi ý mẫu thời trang cho người dùng, giúp cho người dùng chủ động hơn trong việc lựa chọn bộ trang phục ưa thích, cũng như có thêm nhiều lựa chọn mà không tốn nhiều thời trang thì cần có một hệ thống AI/Machine learning tích hợp trong hệ thống website hoặc ứng dụng di động để thực hiện việc này.

1. **Mục tiêu**

Để giải quyết bái toán tạo câu bình luận cho ảnh thời trang, nhiều mô hình đã được nghiên cứu và không ngừng cải tiến. Gần đây nhất là bài viết “Fashion Coordinates Recommender System using Photographs from Fashion Magazines” [1] của Tomoharu Iwata và cộng sự, “Hi, Magic Closet, Tell Me What to Wear!” [2] của Si Liu và cộng sự đề xuất tập trung vào gợi ý những mẫu thời nhưng vẫn còn hạn chế về dữ liệu và xu hướng đa dạng trong việc mua sắm của người dừng. Bài viết “Explainable Fashion Recommendation with Joint Outfit Matching and Comment Generation” [3] do Yujie Lin, Pengjie Ren và cộng sự được công bố tháng 12 năm 2018 đề xuất Neutral Fashion Recommendation framework (NFR), mô hình gợi ý những mẫu trang phục phối với nhau và từ các mẫu trang phục đó phát sinh ra câu bình luận. Mô hình NFR do Yujie Lin, Pengjie Ren và cộng sự được công bố có nhiều ưu điểm về mặt cải tiến độ chính xác theo bảng so sánh kết quả với những mô hình nghiên cứu trước đây được công bố [3] về cả chức năng phối trang phục và phát sinh câu bình luận. Tuy nhiên, NFR vẫn còn nhiều mặt hạn chế về mặt bộ dữ liệu và câu bình luận phát sinh chưa thực sự hiệu quả.

Vậy mục tiêu của đề tài này là thực hiện lại mô hình NFR để áp dụng vào bài toán tạo câu bình luận cho ảnh thời trang.

1. **Phạm vi đề tài**

Trong đề tài này, dựa trên những nghiên cứu Neutral Fashion Recommendation framework do Yujie Lin, Pengjie Ren và cộng sự đề xuất, nhóm tập trung vào nhiệm vụ phát sinh câu bình luận cho một bộ trang phục của mô hình NFR, từ đó đề xuất ra cải tiến cho mô hình NFR để đạt được câu bình luận chính xác hơn.

# Chương II. Tổng quan về học sâu (Deep Learning)

1. **Giới thiệu về Deep Learning**

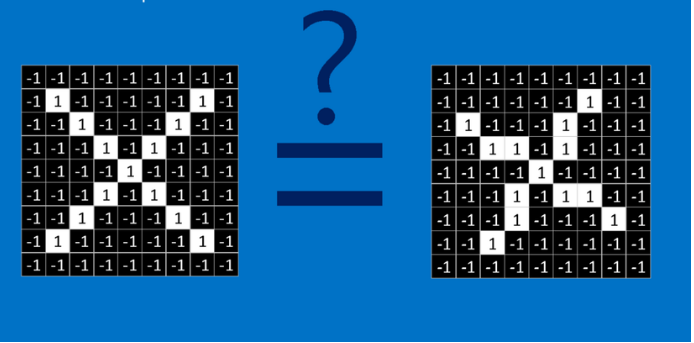
Deep Learning là một phần của ngành [máy học](https://vi.wikipedia.org/wiki/M%C3%A1y_h%E1%BB%8Dc) dựa trên một tập hợp các thuật toán để cố gắng mô hình dữ liệu trừu tượng hóa ở mức cao bằng cách sử dụng nhiều lớp xử lý với cấu trúc phức tạp, hoặc bằng cách khác bao gồm nhiều biến đổi phi tuyến. [5]

Các nghiên cứu trong lĩnh vực Deep Learning cố gắng thực hiện các đại diện (representation) tốt hơn và tạo ra các mô hình để tìm hiểu các đại diện (representation) này từ dữ liệu không dán nhãn quy mô lớn [5]. Một số đại diện được lấy cảm hứng bởi những tiến bộ trong [khoa học thần kinh](https://vi.wikipedia.org/wiki/Khoa_h%E1%BB%8Dc_th%E1%BA%A7n_kinh) và dựa trên các giải thích của mô hình xử lý và truyền thông tin trong một hệ thống thần kinh của con người.

Nhiều kiến trúc của Deep Learning đã được nghiên cứu như mạng neuron sâu (deep neural networks), mạng neuron tái phát (recurrent neural networks), mạng neuron tích chập (convolutional neural networks) và ứng dụng vào nhiều lĩnh vực như thị giác máy tính (computer vision), nhận dạng giọng nói (speech recognition), xử lý ngôn ngữ tự nhiên (natural language processing), phân tích hình ảnh y tế (medical image analysis), v.v. Những ứng dụng của Deep Learning được chứng minh đã tạo ra kết quả tương đương và trong một số trường hợp vượt trội so với con người.

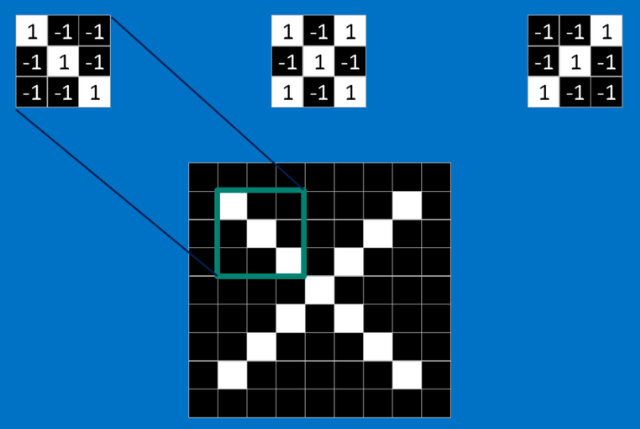
1. **Mạng tích chập (Convolutional Neural Network)**

Convolutional Neural Network (CNN hoặc ConvNet) là một lớp các mạng lưới thần kinh sâu [6], được ứng dụng hầu hết phổ biến để phân tích hình ảnh thị giác (visual imagery) như nhận diện hình ảnh và video (image and video recognition), phân loại ảnh (image classification), phân tích ảnh y tế (medical image analysis).



***Hình 1****: Minh họa đơn giản nguyên lý hoặt động của CNNs* [36]

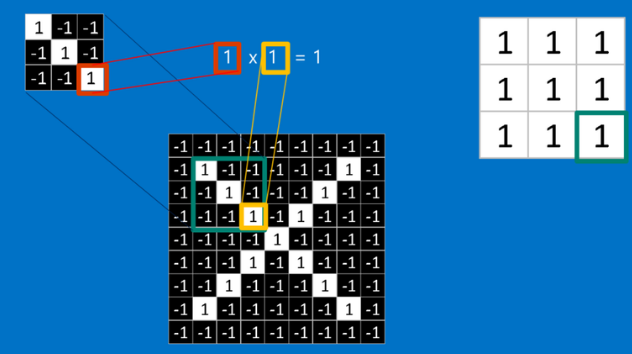
**2.1. Đặc trưng (Feature)**



***Hình 2:*** *Cách so sánh trong CNN*. [36]

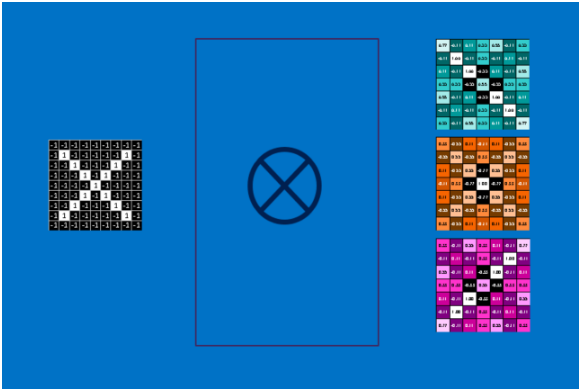
Trong CNN việc so sánh này sẽ chia nhỏ ra từng mảnh để so sánh, các mảnh này gọi là đặc trưng (feature), có kích thước là một mảng 2 chiều. Bằng cách tìm ở mức thô các feature khớp nhau ở cùng vị trí trong hai hình ảnh, CNNs nhìn ra sự tương đồng tốt hơn nhiều so với việc khớp toàn bộ bức ảnh.

* 1. **Tích chập (Convolutional)**



***Hình 3:*** *Nguyên lý tạo ra feature map*

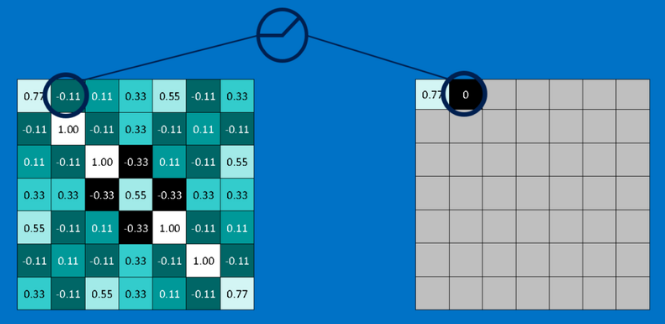
Khi xem một hình ảnh mới, CNN không biết chính xác nơi các feature này sẽ khớp nên nó sẽ thử chúng khắp mọi nơi. Khi tính toán sự khớp của một feature trên toàn bộ ảnh, chúng ta làm thành một filter (bộ lọc), để tính toán sự khớp này ta sẽ nhân mỗi điểm ảnh trong feature với giá trị điểm ảnh tương ứng trong mảnh hình ảnh được so sánh, rồi tính trung bình cộng (cộng tổng lại và chia cho số lượng điểm ảnh trong feature). Nếu tất cả các điểm ảnh đều khớp, ta cộng lại rồi chia trung bình cộng thì sẽ bằng 1.



***Hình 4:*** *Thực hiện toàn bộ quá trình tích chập cho từng feature khác.*

Kết quả là tập hợp các hình ảnh đã được lọc (feature map), mỗi cái ứng với filter.

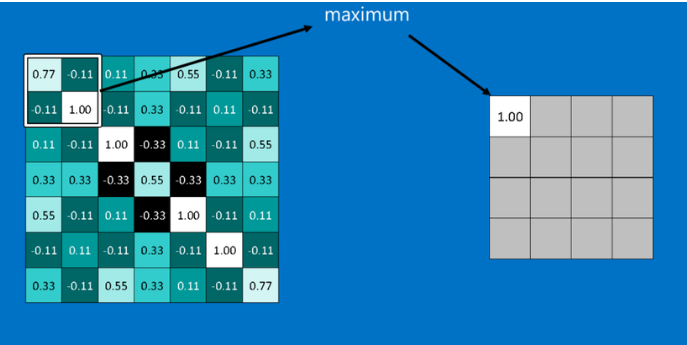
Rectified Linear Unit (ReLU) là hàm activation (activation function) dùng để tinh chỉnh lại các đơn vị tuyến tính, cụ thể là bất cứ nơi nào có số âm thì hoán đổi nó với 0.



***Hình 5:*** *ReLU activation function*

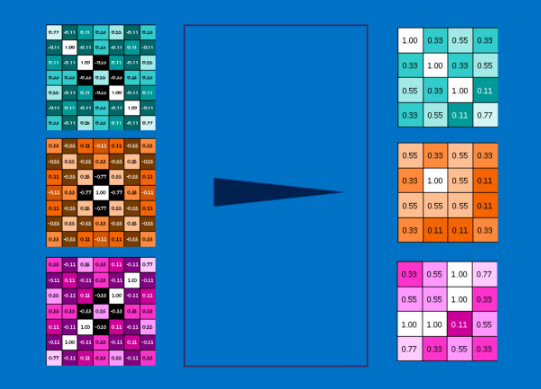
* 1. **Pooling**

Pooling là một cách lấy những hình ảnh lớn và làm co chúng lại trong khi vẫn giữ các thông tin quan trọng nhất trong đó.



***Hình 6:*** *Cách hoạt động của Pooling.* [36]

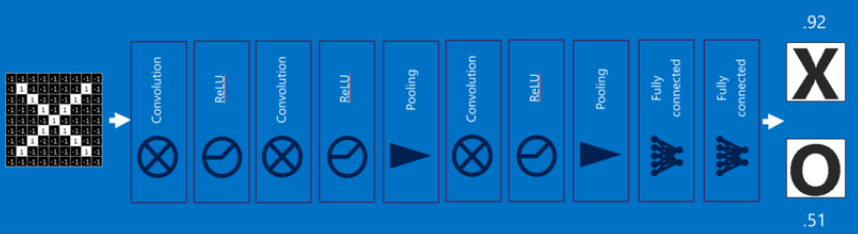
Duyệt bước một ô vuông cửa sổ nhỏ dọc trên một hình ảnh và lấy giá trị lớn nhất từ cửa sổ ở mỗi bước. Sau khi pooling, một hình ảnh sẽ có khoảng một phần tư số điểm ảnh so với lúc bắt đầu, vì những giá trị lớn thể hiện mức độ khớp với đặc trưng (feature) của ảnh đã được lấy.



***Hình 7:*** *Thu gọn những feature map quan trọng sau khi qua pooling.*

* 1. **Fully connected layers**

Ở 2 giai đoạn trước là Convolution và Pooling đã tạo ra một tập các hình ảnh đã được lọc và thu gọn. Các feature map trở nên lớn hơn và phức tạp hơn, và hình ảnh trở nên nhỏ gọn hơn. Điều này giúp các layer thấp hơn đại diện cho các khía cạnh đơn giản của hình ảnh, chẳng hạn như các cạnh và các điểm sáng. Các layer cao hơn có thể đại diện cho các khía cạnh tinh vi hơn của hình ảnh, chẳng hạn như các hình khối (shape) và các hình mẫu (pattern).



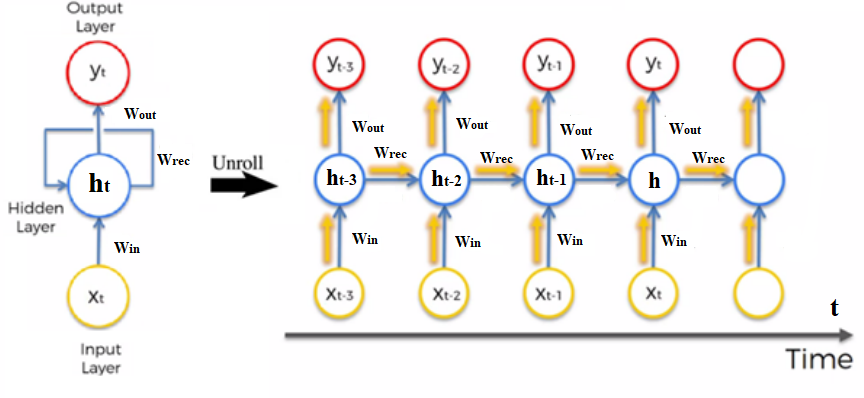
***Hình 8:*** *Miêu tả quy trinh các lớp của CNN.* [36]

Các layer được kết nối đầy đủ (Fully connected layer) lấy các hình ảnh đã lọc ở cấp cao và chuyển chúng thành node mang trọng số (weight). Các layer được kết nối đầy đủ là một khối chính của mạng nơ-ron truyền thống (ANN). Thay vì coi đầu vào như một mảng hai chiều, chúng được coi như một list đơn và tất cả đều được xử lý giống nhau. Một số giá trị cho biết hình ảnh là một X tốt hơn nhiều so với những giá trị khác và một số lại đặc biệt tốt khi cho biết hình ảnh là một O. Chúng được thể hiện như trọng số (weight) hoặc là mức độ kết nối giữa mỗi giá trị (trong list) và mỗi loại (X hay O).

1. **Mạng hồi quy (Recurrent Neural Network)**
   1. **RNN (Recurrent Neural Network)**

RNN (Recurrent Neural Network) là một mạng neural sử dụng chủ yếu để xử lý các bài toán ngôn ngữ tự nhiên (Natural Language) và những nhiệm vụ liên quan đến xử lý chuỗi. Ngoài ra do bản chất của RNN là tính toán tuần tự (sequencial computation) nên thích hợp cho những bài toán dự đoán về dữ liệu tuần tự như xử lý giọng nói (speech recognition), tạo ra nhạc (music generation), phân loại câu (sentiment classification), phân tích chuỗi DNA (DNA sequence analysis), dịch ngôn ngữ (Machine translation), nhận diện hoạt động của video (Video activity recognition), v.v. [35]

Với cái nhìn tổng quát, ý tưởng chính của RNN là sử dụng chuỗi các thông tin (dữ liệu chuỗi đầu vào), thực hiện cùng một tác vụ cho tất cả các từ của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Ví dụ như khi xử lý dịch một chuỗi từ ngôn ngữ này sang ngôn ngữ khác. Các từ sẽ được lần lượt dịch từ trái sang phải, lúc này RNN sẽ dự đoán nghĩa của từ tại thời điểm đang thực hiện bằng cách căn cứ vào những từ đã được dịch trước đó. Cơ chế hoạt động này cho thấy rằng RNN có thể nhớ được thông tin qua từng giai đoạn xử lý trước đó, dựa vào đó để có thể tiếp tục xử lý cho giai đoạn tiếp theo.



***Hình 9:*** *Kiến trúc của mô hình RNN.* [35]

Tại mỗi thời điểm t, từ trong chuỗi được đưa vào hidden layer với ký hiệu là để dự đoán ra được kết quả là . Quá trình này được thực hiện lặp lại (recurrent) cho đến khi dự đoán ra hết tất cả các từ trong câu. Hidden layer được xem như là “memory”, lưu lại những thông tin ở hidden layer trước đó để dự đoán cho từ, và sau đó cập nhật thông tin hiện tại để tiếp tục cho giai đoạn dự đoán tiếp theo. Trong đó hidden layer ( tại thời điểm t được tính bằng công thức.

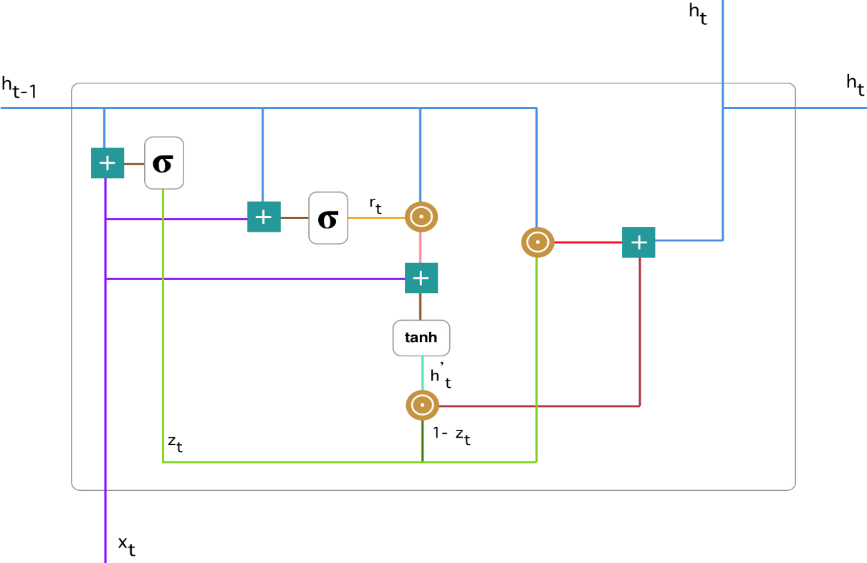
Với là các ma trận trọng số. Lấy tổng tích vô hướng (dot-product) của từ với ma trận trọng số và hidden layer ở thời điểm t-1 với ma trận trọng số . Tổng này sẽ đưa qua hàm , đây là một hàm phi tuyến tính như tanh hay ReLU, nhằm giúp cho tinh chỉnh lại sự chênh lệch quá lớn giữa các trọng số của hidden layer tại thời điểm t sau khi thực hiện phép tính về với miền từ khoảng (-1,1).

Từ dự đoán (tại thời điểm t được tính bằng cách lấy hidden layer hiện tại nhân tích vô hướng (dot-product) với ma trận trọng số , đưa qua hàm softmax giúp chuẩn hóa (normalization) lại phân bổ xác suất về miền giá trị. [0,1]

Tại mỗi thời điểm t, điểm mất mát (loss) giữa từ thực sự (actual output) và từ được dự đoán (predicted output) được tính và tổng hợp để cập nhật lại trọng số cho các ma trận . Mất mát (loss) này được lan truyền ngược lại cập nhật những trọng số để tối giản mất mát, kỹ thuật này gọi là Back propagation through time (BPTT).

Mạng neural tái phát (RNN) có nhược điểm khi phải đối mặt với vấn đề hao hụt hoặc bùng nổ đạo hàm (Vanishing or Exploding Gradient). Khi xử ký chuỗi có kích thước lớn, lặp lại phép nhân của ma trận trọng số nhiều lần để dự đoán các từ ở chuỗi (long term) khiến cho RNN dễ đối mặt với hiện tường hao hụt hoặc bùng nổ đạo hàm (Vanisshing/Explodig gradient). [35]

* 1. **GRU (Gated Recurrent Unit)**

****

***Hình 10:*** *Gated Recurrent Unit (GRU)*

- Reset Gate: lượng thông tin ở những giai đoạn xử lý trước đó cần bỏ.

- Update Gate: xác định bao nhiêu thông tin xử lý ở các giai đoạn trước và hiện cần được truyền qua cho giai đoạn xử lý tiếp theo.

Từ đó mô hình có thể quyết định lấy tất cả thông tin từ quá khứ và loại bỏ nguy cơ vấn đề hao hụt đạo hàm (vanishing gradient problem).

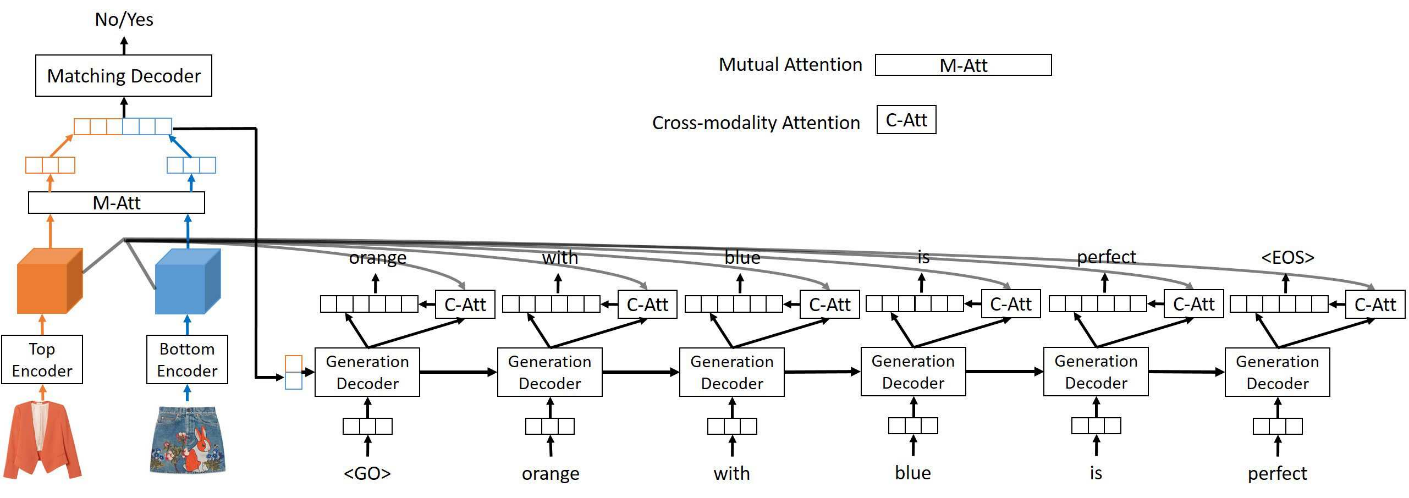
# Chương III. Phương pháp đề xuất

1. **NFR framework**

Neutral Fashion Recommendation [3] là một Novel Neural Network framework do Yujie Lin, Pengjie Ren và cộng sự đề xuất, cung cấp gợi ý thời trang và tạo ra câu bình luận trừu tượng cho bộ trang phục được phối. NFR có 2 nhiệm vụ chính bao gồm: phối các bộ trang phục cho phù hợp và từ những bộ trang phục phù hợp phát sinh câu bình luận để miêu tả cảm xúc đối với bộ trang phục.

Để phối các mẫu trang phục, dữ liệu đầu vào là một hình ảnh của trang phục phần trên (như áo sơ mi, áo thun, áo len, v.v), nhiệm vụ lúc này của hệ thống sẽ gợi ý một danh sách các trang phục phần dưới (như váy, quần jean, quần thun, v.v) từ bộ sưu tập để chọn ra những trang phục phần dưới có độ phù hợp nhất với trang phục phần trên được cung cấp. Đồng thời phát sinh ra một câu bình luận cho mỗi bộ trang phục có độ phù hợp nhất được phối, nhằm thể hiện được cảm xúc đối với bộ trang phục.

1. **Kiến trúc của NFR framework**



***Hình 11****: Sơ đồ kiến trúc tổng quan của neural fashion recommendation (NFR)* [3]

NFR gồm có 3 phần chính:

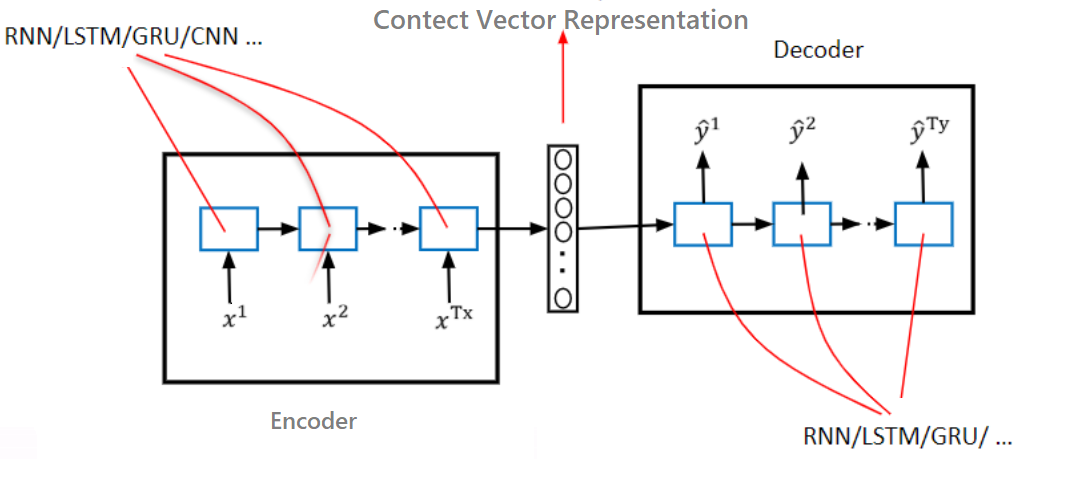
- Giai đoạn Encoder: trích xuất đặc tính (feature) của ảnh trang phục trên và dưới thành các latent vector bằng cách sử dụng convolutional neural network (CNN).

- Giai đoạn Decoder cho nhiệm vụ phối trang phục: sử dụng cơ chế Mutual Attention [4] để lấy chéo đặc tính của trang phục trên và dưới từ đó trích xuất ra được các đại diện vector (representation vectors) của trang phục trên và dưới mà đặc tính (feature) chéo của nhau. Các đại diện vector (representation vector) là nguyên liệu cho quá trình phối trang phục với nhau (Matching Decoder).

- Giai đoạn Decoder cho nhiệm vụ phát sinh câu bình luận: các đại diện vector (representation vectors) được đưa vào quá trình phát sinh câu bình luận dành cho cặp ảnh thời trang. Trong quá trình tạo ra các từ cho câu bình luận, cơ chế Cross-modality Attention [4] được áp dụng để lấy đặc tính (feature) của từ và đặc tính (feature) của cặp ảnh nhằm tăng độ chính xác cho bình luận đối với cặp ảnh thời trang.

1. **Encoder – Decoder Architecture**

Kiến trúc Encoder – Decoder được chia ra làm 2 giai đoạn chính: mã hóa (encoder) và giải mã (decoder). Cả Encoder và Decoder đều là một mạng lưới (network) có thể được sử dụng độc lập nhau. Nhưng nhiệm vụ sẽ khác nhau, Encoder network được sử dụng để ánh xạ dữ liệu thành các đại diện đặc trưng và Decoder network sử dụng đại diện đặc trưng này như là dữ liệu đầu vào và thực hiện việc dự đoán và ra quyết định.

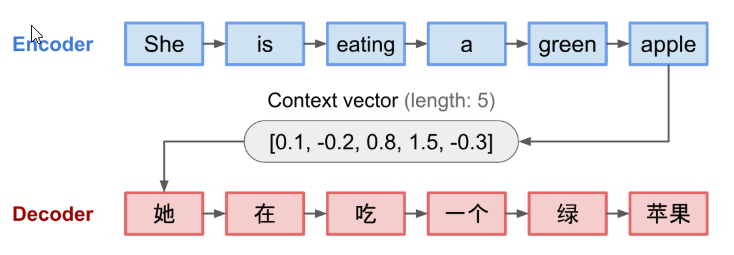
***Hình 12:*** *Sơ đồ kiến trúc Encoder – Decoder Architecture*

Cụ thể hơn, mã hóa (encoder) là có thể là ứng dụng mạng tích chập (Convolutional Neural Network) hoặc mạng mạng hồi quy (Recurrent Neural Network) để thực hiện nhiệm vụ trích xuất đặc tính của dữ liệu thô (features) thành một vector đại diện có độ dài cố định gọi là context vector representation.

Giải mã (decoder) sử dụng vector đại diện có độ dài cố định (context vector representation) để làm dữ liệu đầu vào bằng cách ứng dụng những mô hình mạng lưới (network) để đưa ra kết quả như ứng dụng mạng hồi quy (Recurrent Neural Network) để đưa ra kết quả.

1. **Sequence to Sequence Model**

Sequence to Sequence là một mô hình ứng dụng kiến trúc Encoder – Decoder, bao gồm 2 giai đoạn mã hóa (encoder) và giải mã (giải mã). Mô hình được ứng dụng vào lĩnh vực xử lý ngôn ngữ tự nhiên, tập trung vào việc biến đổi một câu được đưa vào (input source) thành một câu mới (target source) và cả 2 câu đều có độ dài tùy ý. [11]

***Hình 13:***Encoder-decoder model, dịch câu “she is eating a green apple” sang tiếng Trung. [11]

Một câu (input source) được đưa vào quá trình encoder để trích xuất đặc trưng thành vector đại diện (context vector representation) có độ dài cố định (fixed length) đại diện cho toàn bộ câu văn được đưa vào (input source). Mô hình Sequence to sequence sử dụng mạng tái phát (Recurrent Neural Networks) cho giai đoạn mã hóa (encoder) và giải mã (decoder), cụ thể là những mô hình biến thể đã được cải tiến của RNN như Long Short Term Memory networks (LSTM) [11] hoặc Gated Recurrent Unit (GRU). [12]

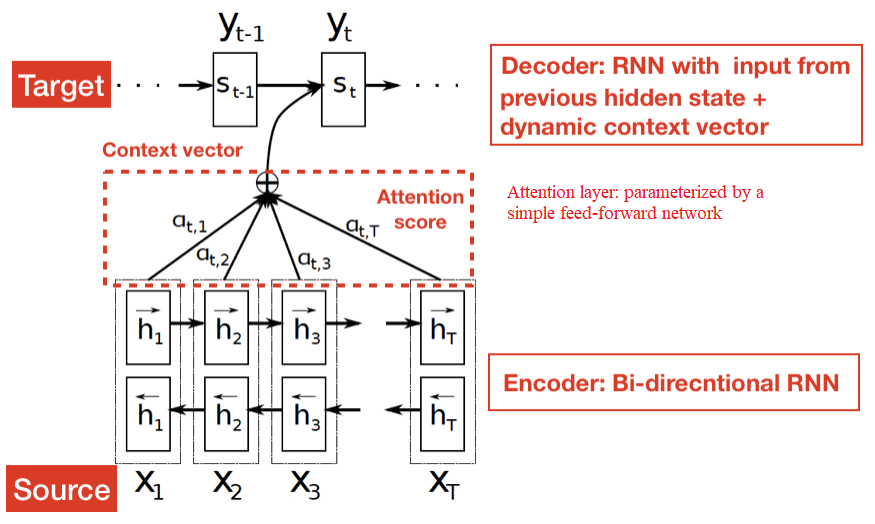
Sequence to sequence model có nhược điểm khi toàn bộ thông tin được nén lại ở vector đại diện (vector context representation) chứa nội dung của toàn bộ câu đưa vào (input source), dẫn đến việc mất thông tin nếu như độ dài của dữ liệu câu đưa vào quá lớn khiến giải mã (decoder) khó có kết quả chính xác (target source). Nguyên nhân, do giai đoạn encoder là một mạng hồi quy (recurrent network) như RNN, hoặc biến thể được cải tiến như LSTM, GRU, dữ liệu phần đầu sẽ bị loãng (diluted downstream) sau khi hoàn thành mã hóa (encoder) toàn bộ đầu vào. Để khắcc phục nhược điểm của mô hình Sequence to sequence, cơ chế chú ý (Attention Mechanism) được Dzmitry Bahdanau, KyungHyun Cho và Yoshua Bengio đề xuất. [15]

1. **Attention Model**

Mô hình chú ý (Attention model) là một mở rộng (extension) cho mô hình Encoder – Decoder được DzmitryBahdanau và cộng sự đề xuất [15]. Từ nhược điểm của mạng tái phát (Recurrent Neural Networks) truyền thống với vấn đề Vanishing Gradient [13], mặc dù để khác phục nhược điểm này nhiều biến thể của RNN như LSTM [15], GRU [12] được đề xuất nhưng vẫn chưa thực sự đạt kết quả tốt khi áp dụng vào mô hình kiến trúc encoder – decoder.

Lấy ý tưởng từ đó, Cơ chế Attention (Attention Mechanism) được Dzmitry Bahdanau và cộng sự [15] đề xuất để mở rộng cho mô hình Encoder – Decoder. Cơ chế chú ý (Attention Mechanism) cho phép mô hình có thể chú trọng vào những phần quan trọng, cụ thể là mức độ liên kết của từ ở dữ liệu đầu vào (input source) và từ sẽ được dự doán (target).

Giai đoạn decoder, mỗi thời điểm tạo ra một từ, cơ chế chú ý (Attention Mechanism) sẽ tìm kiếm một tập hợp các vị trí trong câu nguồn (source sentence) nơi thông tin quan trọng nhất được chú ý [15], việc lựa chọn này thực chất là một ma trận trọng số (weighted matrix) thể hiện xác suất chú ý (attention probability) giữa mỗi dữ liệu nguồn (source token) và từ sẽ được dự đoán (target), nhằm hạn chế hidden state tại giai đoạn decoder bị loãng thông tin hay mất thông tin ở dữ liệu nguồn. Thay vì chỉ sử dụng layer cuối cùng của giai đoạn Encoder như một vector đại diện đặc trưng (context vector representation) cho toàn bộ nội dung dữ liệu đầu vào (input source), thì lúc này với vector ngữ cảnh (context vector/attention vector) sẽ được tạo ra với nội dung là những thông tin quan trọng của từ ở nguồn (source) có mối liên hệ với từ cần dự đoán (target). Context vector lúc này sẽ hiệu chỉnh lại thông tin mà hidden state t-1 ở giai đoạn decoder đang ghi nhớ để khác phục nhược điểm của mô hình mạng tái phát (recurrent neuron network).



***Hình 14:*** *Mô hình đang tạo ra từ yt tại thời điểm t bằng cách áp dụng cơ chế Attention*. [15]

Cơ chế chú ý (Attention Machenism) được ứng dụng nhiều để xử lý dịch văn bản (Neural Machine Translation), phát sinh câu chú thích cho ảnh (Image Captioning), tóm tắt văn bản (Text Summarizarion), v.v [16]

Với kiến trúc chú ý (Attention model) [15], xác suất để phát sinh ra từ dự đoán tại thời điểm i là

p ( |, . . . , , x) = g(, , ) (1)

Với là hidden state của RNN tại thời điểm i được tính bằng

= (2)

Khác với những mô hình trước đó, tại mỗi thời điểm i, mô hình chú ý (Attention model) sẽ tính xác suất dựa trên sự khác biệt của vector ngữ cảnh (context/attention vector) đối với mỗi từ được dự đoán . Vector ngữ cảnh phụ thuộc vào chuỗi các hidden state ở giai đoạn mã hóa (encoder) như hình 5. Với mỗi hidden state ở giai đoạn mã hóa (encoder) chứa thông tin toàn bộ của giữ liệu đầu vào (input sequence), và sẽ tập trung vào từ được đưa vào để mã hóa tại thời điểm . Chính vì vậy mà vector ngữ cảnh chứa tổng các trọng số (alignment score) của các hidden state tại thời điểm j đối với từ được dự đoán, để thể hiện “how well the inputs around position j and the output at position i match” [15]

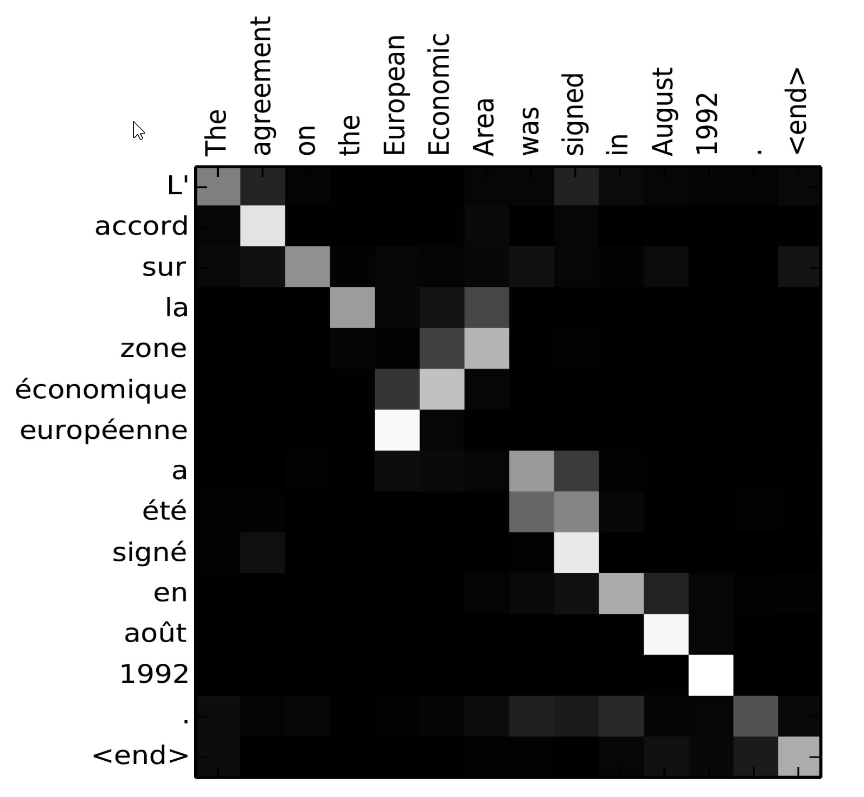
(3)

Trọng số (alignment score) cho mỗi hidden state tại thời điểm j được tính bằng cách áp dụng hàm Softmax vào alignment model để normalize các tất cả các điểm (scores) tạo ra phân phối xác suất có điều kiện đối với từ được dự đoán.

(4)

Với là alignment model với mục đích đánh giá mức độ tương quan của từ (input source) tại thời điểm j của giai đoạn mã hóa (encoder) với từ dự đoán (output source) tại vị trí i.

(, )



***Hình 15:*** *Mức độ tương quan (correlation) giữa dữ liệu nguồn (source) và dữ liệu dự đoán (target).* [15]

Cột ngang và cột dọc thể hiện những từ trong câu nguồn (source) và những từ được dự đoán (target).

Việc áp dụng cơ chế (Attention Mechanism) là một cơ chế giúp mô hình có thể tập trung vào các phần quan trọng trên dữ liệu, bằng việc tạo 1 alignment model ( để tính các alignment score ). Xác suất alignment score ) tại j phản ánh sự quan trọng của hidden state ) của giai đoạn encoder, với hidden state () trong việc quyết định hiden state tiếp theo ) và từ dự đoán (. Có nhiều cách để chọn alignment model như Content Base Attention [17], Additive Attention [15], Multipliction Attention, Dot Product [18].

Nhược điểm của mô hình chú ý (Attention model) là tính toán vector ngữ cảnh (context/attention vector) thể hiện những thông tin tương quan quan trọng của giữa câu nguồn (source sentence) và câu dự đoán (target sentence) để hiệu chỉnh cho decoded states, mà thông tin chú ý (attention information) ở bản thân câu nguồn (source sentence) và ở câu dự đoán (target sentence) bị bỏ qua. Thêm vào đó mạng tái phát (RNN) và các biến thể của RNN thì tính toán theo kiểu tuần tự (sequential computation) dẫn đến việc các hidden state sẽ phụ thuộc lẫn nhau và khó có thể tính toán song song (parallelization), không hiệu quả về cả mặt thời gian [20] [21]. Cuối cùng là vấn đề **Long Range Dependencies [22] mà RNNs vẫn tồn tại mặt dù có những biến thể cải tiến được đề xuất.**

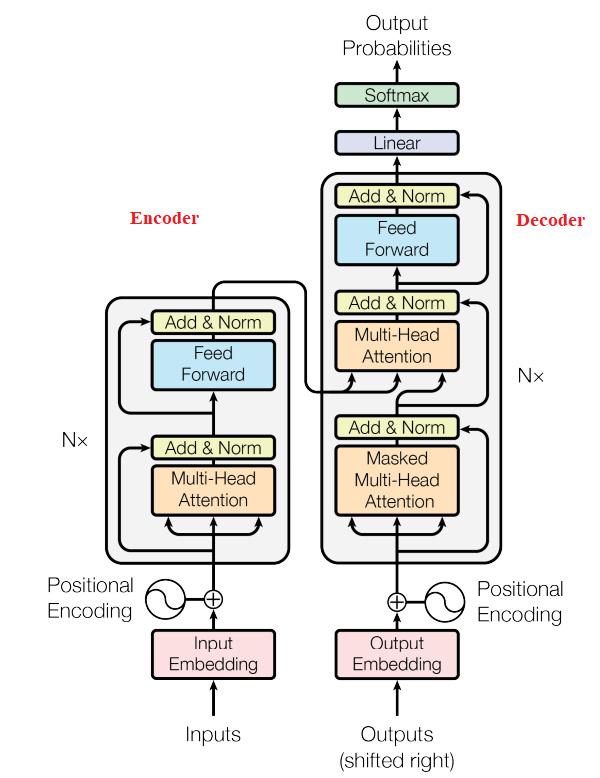
1. **Transformer Architecture**

Tranformer là một kiến trúc mạng lưới (network architecture) dựa hoàn toàn trên cơ chế chú ý (attention mechanism) do Ashish Vaswani và cộng sự đề xuất [27]. Kiến trúc này khác với Sequence to Sequence Model trước đây, vì đã ứng dụng cơ chế Seft attention và loại bỏ mạng hồi quy (Recurrent Neural Network).

Mục tiêu của kiến trúc Transformer là có thể giúp cho mô hình tính toán song song (Parallelization of Segquence to Sequence) rút ngắn lượng lớn thời gian trong quá trình học máy (trainning) và áp dụng cơ chế Seft attention để chú ý đến những vị trí khác nhau của một chuỗi đầu vào (input) từ đó tính toán thành đại diện (representation) cho chuỗi đó. Tiêu chí cải tiến của Transformer đưa ra những khắc phục ở mặt hạn chế được đề cập ở mô hình Attention Model.

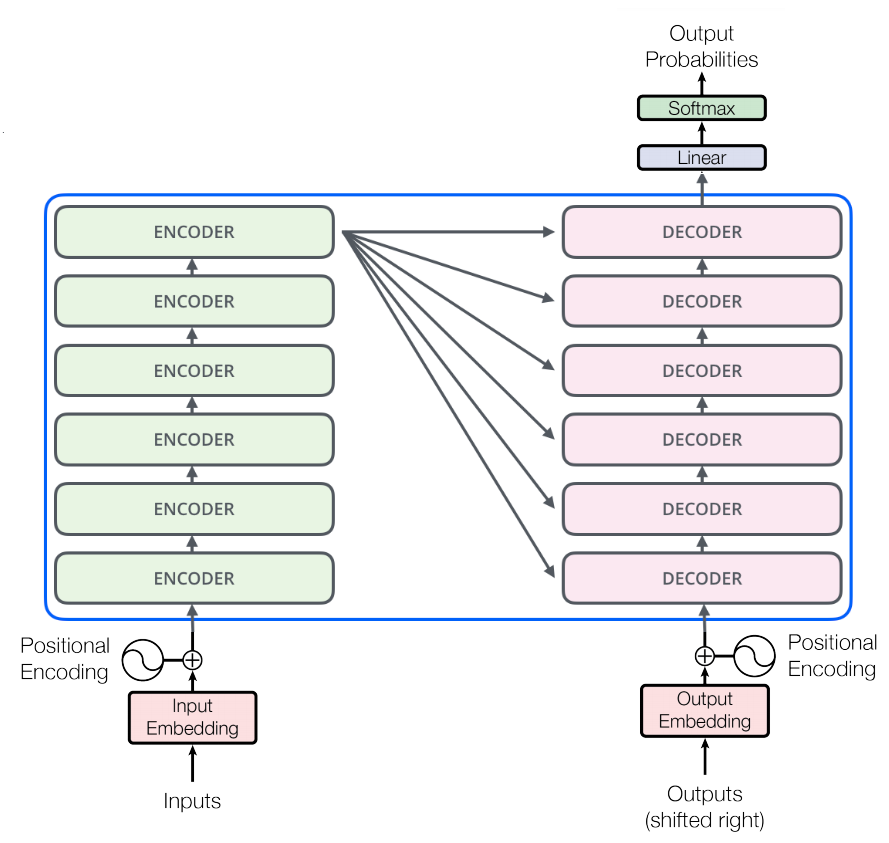
* 1. **Mô hình kiến trúc của Transformer**

Kiến trúc encoder – decoder vẫn được áp dụng cho mô hình, giai đoạn encoder ánh xạ đại diện chuỗi đầu vào thành một chuỗi đại diện tiếp theo . Chuỗi đại diện z sẽ được đưa vào giai đoạn decoder để tạo thành chuỗi dự đoán với mỗi từ tại 1 thời điểm. Tại mỗi bước, mô hình sẽ tự động hồi quy (auto-regressive), và sử dụng các từ được tạo trước đó làm đầu vào bổ sung khi tạo các từ tiếp theo. [27]



***Hình 16:*** *Kiến trúc mô hình Transformer.* [27]

Nhìn tổng thể kiến trúc của Transformer, cả encoder và decoder các mô-dun được xếp chồng lên nhau (stack of encoders/decoders) nhiều lần, số lượng khối xếp chồng của cả 2 giai đoạn encoder và decoder đều bằng nhau, được mô tả bởi ký hiệu Nx trong hình 16. Mỗi khối encoder/decoder được cấu tạo từ 2 loại lớp con (sub-layer) gồm seft-attention và position-wise fully connected Feed Forward Neural Network (FFN).



***Hình 17:*** *Các khối encoder/decoder được xếp chồng nhau trong mô hình Transformer.* [31]

* 1. **Input Embedding và Positional Embedding**

Đây là bước quan trọng để mã hóa thông tin theo trình tự các từ của chuỗi ở dữ liệu đầu vào. Trước tiên dữ liệu đầu vào sẽ được mã hóa thành các mã thông báo (token). Mã thông báo (tokenizer) mã hóa chuỗi bằng cách ngắt chuỗi thành từ khóa phụ nếu từ đó không có trong tập dữ liệu và Thêm mã thông báo bắt đầu và kết thúc vào đầu vào và mục tiêu. Sau đó chuyển đổi mã thông báo dữ liệu đầu vào (input tokens) và mã thông báo dữ liệu dự đoán (output tokens) của từng từ thành những vec-tơ không gian n chiều (n-dimensional space) gọi là input embedding vector.

Tiếp theo mã hóa vị trí (positional encoding) các từ khác nhau trong câu sẽ được thêm vào input embedding vector, mã hóa vị trí (positional encoding) có cùng không gian chiều (n-dimentional space) với input embedding. Bởi vì mô hình ko sử dụng mạng tái phát (recurrent neural network) hoặc mạng tích chập (convolutional neural network) có thể nhớ được từng từ trong chuỗi đưa vào mô hình như thế nào, việc vị trí (positions) được thêm vào biểu diễn nhúng (embedding representation) cung cấp cho mỗi từ trong chuỗi thông tin về vị trí tương quan (relative position) của các từ trong chuỗi. Bởi vì một chuỗi phụ thuộc vào trình tự các từ trong chuỗi đó [27] [29] [30]

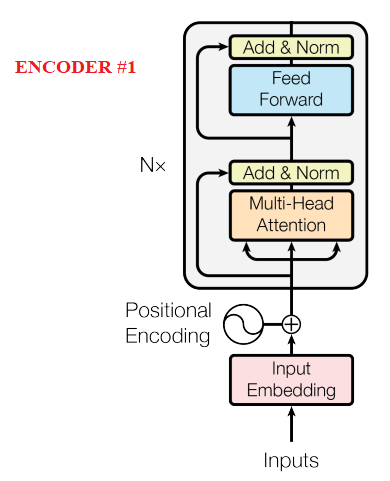
Hàm sine và cosine được đề xuất trong mô hình Transformer [27] cho việc tính toán mã hóa vị trí (positional encoding)

Với pos là vị trí và i không gian chiều (dimension).

* 1. **Giai đoạn Encoder**

Giai đoạn này bao gồm nhiều lớp encoder chồng nhau (stack of encoder layers), hình 18 bên dưới miêu tả một lớp encoder. Mỗi lớp encoder sẽ có 2 lớp con (sub-layer) xử lý bên trong gồm có: Multi-Head Self-Attention và position-wise fully connected feed-forward network.

Giữa 2 sub-layer là một lớp chuẩn hóa (layer normalization) sẽ được áp dụng residual connection trước khi chuẩn hóa. Để tạo điều kiện cho các kết nối còn lại này (residual connection), tất cả các lớp con (sublayer) trong mô hình, cũng như các lớp nhúng (embedding layers), tạo ra các đầu ra có kích thước dmodel = 512. [27]

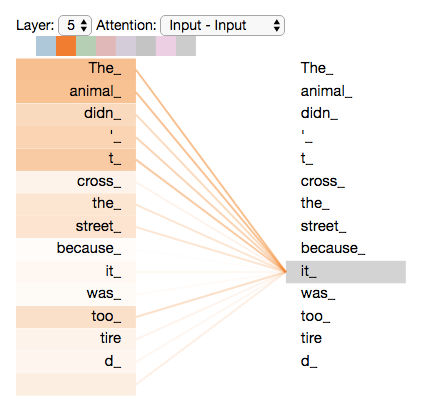


***Hình 18****: Một lớp của giai đoạn encoder.* [27]

* + 1. **Seft-attention trong Transformer**

Ở mức độ tổng quan, Self-attention hoặc Intra-attention [25], [26], [27] là một trường hợp khác của cơ chế chú ý (Attention Mechanism), đây là một cơ chế chú ý đến sự liên quan của các vị trí khác nhau trong cùng một chuỗi (a single sequence) để tính toán biểu diễn (representation) của chuỗi đó.

Để làm rõ hơn về Seft-attention, ta lấy ví dụ một câu nguồn (input senquence) “The animal didn't cross the street because it was too tired”. Trong câu trên ta có thể hiểu được chữ “The animal” và “it” là những từ đồng nghĩa với nhau và được thể hiện ở những vị trí khác nhau trong câu. Seft-attention cho phép nó xem xét các vị trí khác trong chuỗi đầu vào để có thể giúp mã hóa tốt hơn cho từ này.

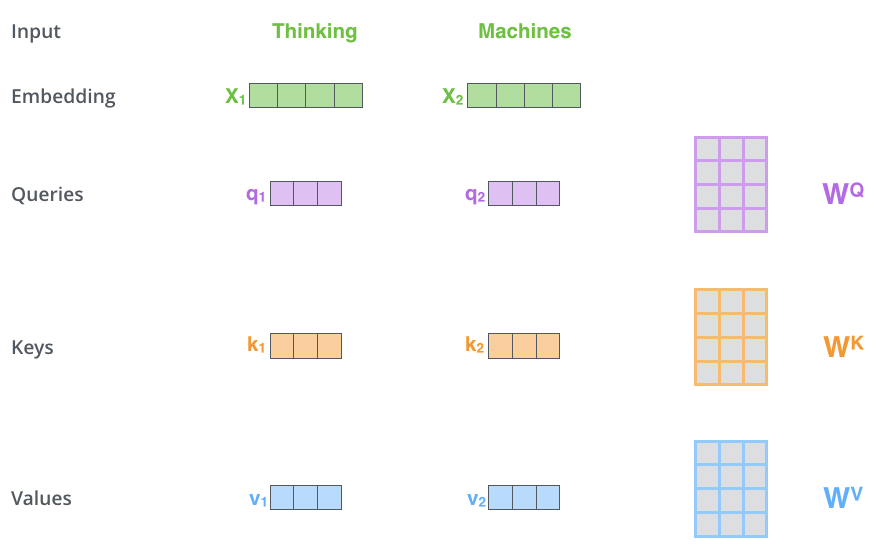


***Hình 19:*** *Mã hóa từ "it" trong encoder layer thứ 5.* [31]

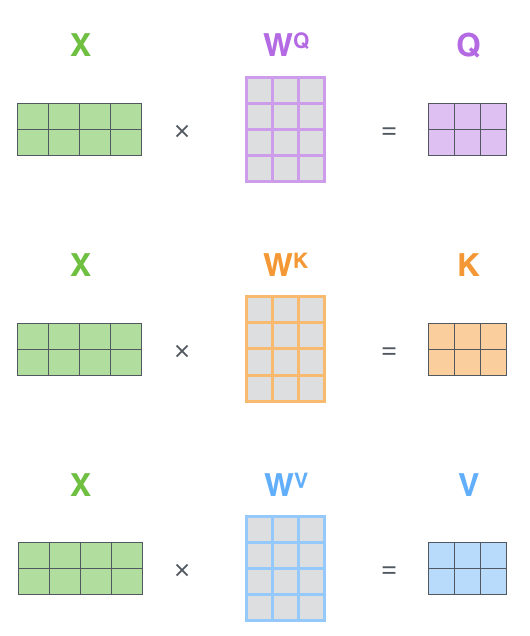
Sau khi mã hóa từ "it" trong encoder layer thứ 5, phần lớn cơ chế chú ý (Seft-attention) đã tập trung vào "The animal" và đưa thông tin mã hóa này vào một phần biểu diễn (representation) của chính nó.

Ở gốc nhìn chi tiết hơn về hàm Seft-Attention được mô tả như ánh xạ một truy vấn (a query) và một tập hợp các cặp khóa (a set of key-value), để tạo ra giá trị đầu ra (output). Trong đó truy vấn (query), khóa (keys), giá trị (values) và giá trị đầu ra (ouotput) đều là các vectơ. Đầu ra (output) được tính là tổng trọng số của các giá trị (values), trong đó trọng số được gán cho mỗi giá trị (value) ược tính bằng hàm tương thích (compatibility function) của truy vấn (query) với khóa tương ứng corresponding key). [27]

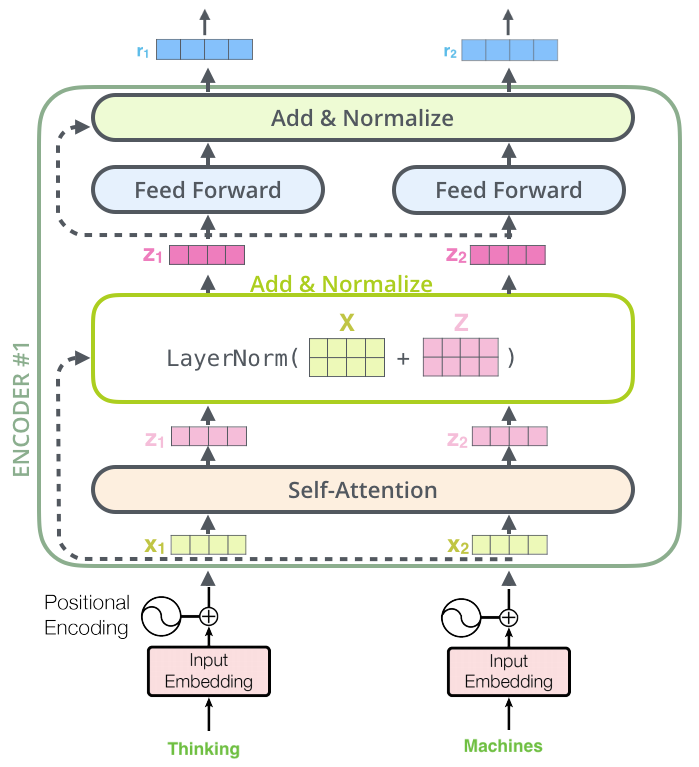
Bước đầu tiên trong việc tính toán sự chú ý (seft attention) là tạo ra ba vectơ (query, key, value) từ mỗi vec-tơ nhúng của từng từ (word embedding vector). Các vectơ này được tạo ra bằng cách nhân vec-tơ nhúng (word embedding vector) với ba ma trận trọng số trong quá trình đào tạo. Vector query, key, value được tạo ra có kích thước không gian (n-dimensional space) là 64 nhỏ hơn vector nhúng của từ (word embedding vector) là 512, kích thước này không thể nhỏ hơn vì đây là kiến trúc lựa chọn để tạo thuận lợi cho việc áp dụng residual connection giữa các sublayer [27].



***Hình 20:*** *Miêu tả việc tạo ra các vector query, key, value cho một từ bằng việc nhân embedding vector với các matix trong số tương ứng.* [31]



***Hình 21:*** *Miêu tả việc tạo ra ma trận query, key, value cho câu dữ liệu nguồn, mỗi hàng trong ma trận X thể hiện 1 từ trong câu dữ liệu nguồn.*



***Hình 22:*** *Giai đoan Encoder ở layer thứ 1, miêu tả việc xử lý các vector đại diện qua từng sublayer*. [31]

Mỗi câu dữ liệu đưa vào (sequence input) sau khi qua giai đoạn input embedding và positional encoding như hình 22, các từ được biểu diễn bằng các vector embedding representation, của số từ với . Sau khi qua Seft attention sublayer, vector đại diện này (vector embedding representation) sẽ được thêm sự liên quan (compositional relationships) của các vị trí khác nhau trong cùng một chuỗi, được ký hiểu bởi z có cùng kích thước với với .Với mỗi vector đại diện (vector embedding representation) tại thời điểm được cập nhật các trọng số (attention score) bằng công thức. [28]

(6)

Với là trọng số (attention score) của từ được chuẩn hóa (normalize) bằng hàm softmax và là value của từ ở vị trí j.

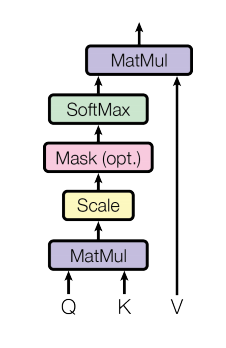
(7)

Và là trọng số (attention score) thể hiện mức độ liên quan giữa vị trí I và j được tính bằng một hàm tương thích (compatibility function). Trong Transformer được Ashish Vaswani và cộng sự đề xuất Scaled Dot-Product Attention để tính toán, được miêu tả ở mục kế tiếp.

* + 1. **Scaled Dot-Product Attention**

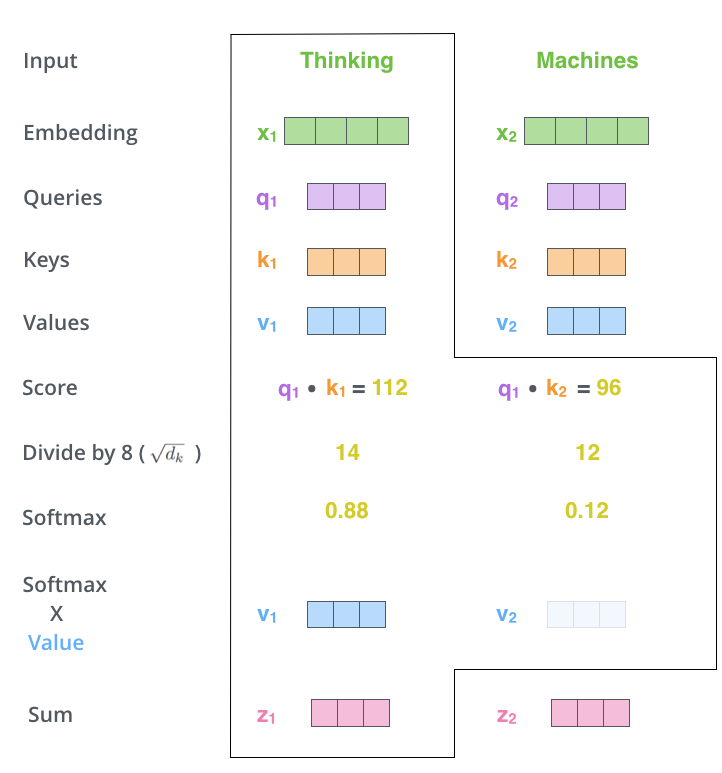
Scaled Dot-Product Attention là một hàm chú ý (attention function) dùng để tính điểm tương thích giữa các từ trong câu. Hàm tính toán trên một bộ truy vấn (a set of queries) thể hiện qua ma trận Q, các khóa (keys) ma trận K với kích thước không gian là và giá trị (values) ma trận V với kích thước không gian là được tính bằng công thức.

Công thức được tính cách lấy tích vô hướng (dot-product) của ma trận query (Q) và ma trận keys (K), chia cho căn bật 2 của không gian ma trận và sử dụng hàm softmax để chuẩn hóa (normalization) để đạt trọng số với giá trị values (V).



***Hình 23:*** *Scaled Dot-Product Attention.* [27]

Lấy ví dụ trên một từ trong chuỗi được đưa vào (input source), sử dụng hàm Scaled Dot-Product Attention để tính điểm chú ý (attention score) của từ “Think” với từ còn lại trong câu như hình 24 bên dưới.



***Hình 24:*** *Tính điểm attention score bằng hàm tương thích Scaled Dot-Product Attention, con số trong hình là minh họa.* [31]

Sau khi có các vector query, value, key của từ “Thinking”, Seft-attention được sử dụng để tính điểm (score) các từ khác trong câu dữ liệu đầu vào (input sentence) đối với từ “Think”, điểm này sẽ chỉ ra rằng mức độ chú ý của từ này đối với các từ khác trong câu dữ liệu đầu vào (input sentence). Điểm chú ý này (attention score) được tính bằng hàm tương thích (compatibility function), cụ thể là Scaled Dot-Product Attention.

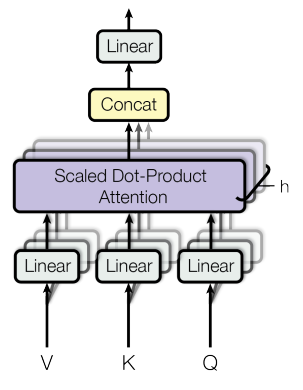
Trước tiên ta tính tích vô hướng (dot product) của vec-tơ và theo mỗi từ tại vị trí trong câu. Lấy kích thước không gian vecto của query, key và score là 64 và không gian vec-tơ word embedding là 512 theo bài nghiên cứu gốc [37]. Tiếp đó ta lấy điểm (score) vừa tính của mỗi từ chia cho 8 vì căn bậc 2 của không gian véc-tơ của query và key là 64. Sau đó chuẩn hóa kết quả này qua hàm Softmax, các trọng số này sẽ luôn nằm trong khoảng từ 0 đến 1.

Cuối cùng, trọng số này sẽ được nhân với các value của từng từ, mục đích để giữ nguyên vẹn các giá trị của các từ mà chúng ta muốn tập trung vào, và làm hạ thấp trọng số của các từ không liên quan (ví dụ bằng cách nhân chúng với các số nhỏ như 0,001). Lúc này ta thu được vec tơ đại diện (vector representation) ký hiệu là cho mỗi từ trong câu, thể hiện được mức độ chú ý của từ tại vị trí đến các vị trí khác nhau trong cùng một chuỗi.

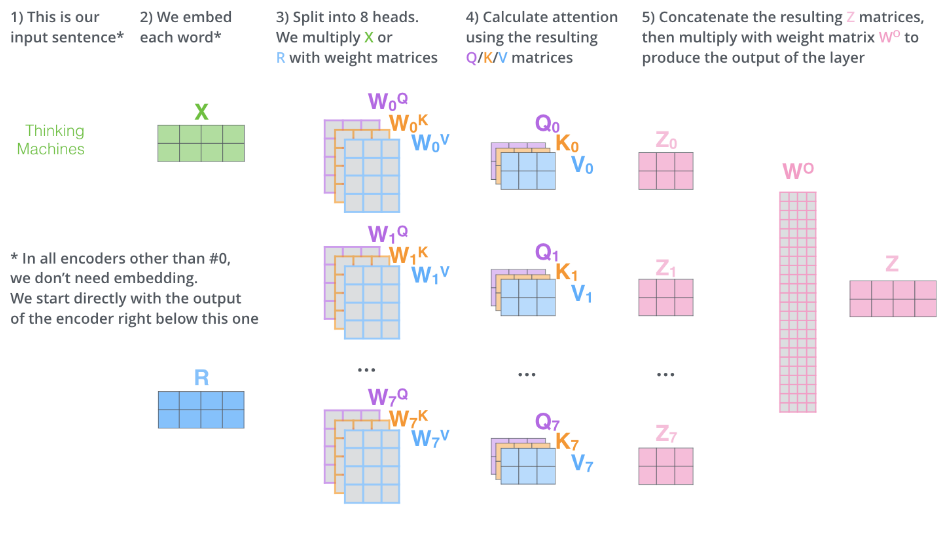
* + 1. **Multi head self attention**

Thay vì chỉ sử dụng một lớp Seft attention, thì Multi head self attention tính toán ra các vec-tơ đại diện tại mỗi thời điểm khác nhau với các ma trận trọng số (weight matrics) khác nhau. Muti-head attention cho phép mô hình tham gia thông tin từ các không gian đại diện khác nhau ở các vị trí khác nhau [27], điều này có nghĩa với một từ thì sẽ có một hoặc nhiều hơn một không gian đại diện (representation subspaces) thể hiện sự đa dạng ngữ nghĩa của từ.

Với

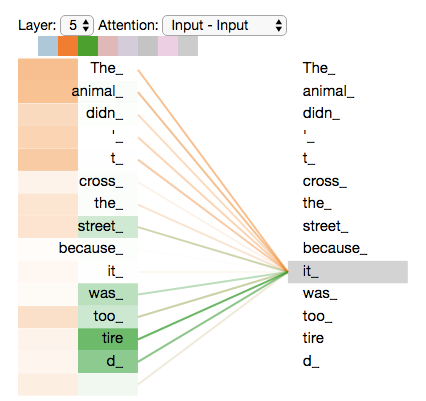


***Hình 25:*** *Multi-Head Attention gồm nhiều lớp h attention chạy song song.*



***Hình 26:*** *Miêu tả chi tiết Multi head Attention xử lý dữ liệu song song qua các multi-head để thành các vector đại diện.*

Với multi-head attention, các ma trận trọng số sẽ khác nhau và tổng hợp ra các vector đại diện khác nhau biểu diễn cho cùng một từ.



***Hình 27:*** *Miêu tả mối quan hệ của từ “it” với từ còn lại trong câu, sau khi đã sử dụng multi-head attention.* [31]

Bởi vì từ ‘it” được mã hóa với multi-head attention nên vector đại diện của từ này được tổng hợp với nhiều không gian vector biểu diễn khác nhau (representation subspaces), dẫn đến một attention head chú ý vào từ “The animal” và một attention khác chú trọng vào từ “tire”.

* + 1. **Position-wise Feed Forward Networks**

The pointwise feed-forward layer nhận giá trị (input) từ multi-head attention layer, và chuyển đổi giá trị qua 2 phép biến đổi tuyến tính (linear transformation) với hàm kích hoạt (ReLU activation) ở giữa. Lớp feed-forward có thể xem là 2 lớp convolution với kernel size =1.

* + 1. **Residual connection**

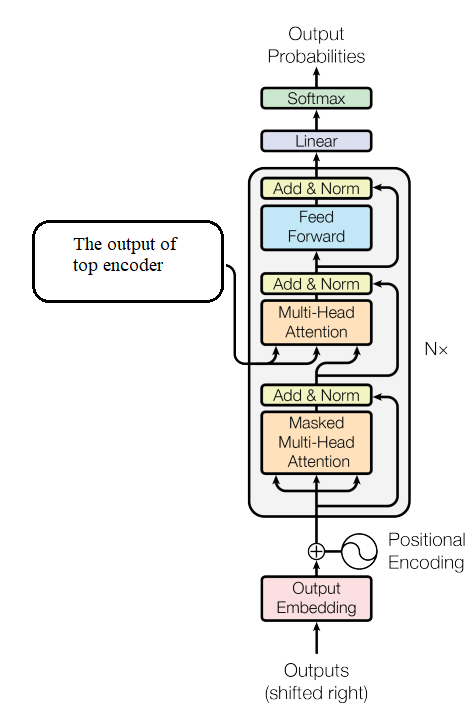
Giữa các sublayer có 1 kết nối gọi là residual connection. Residual connection giúp tránh đi hiện tượng bùng nổ đạo hàm (vanishing gradient) ở mạng sâu (deep networks)

Đầu ra của mỗi sublayer sẽ là:

Với x là vector đại diện dữ lieu đầu vào, Sublayer(x) là vector đại diện mới được tạo khi qua sublayer.

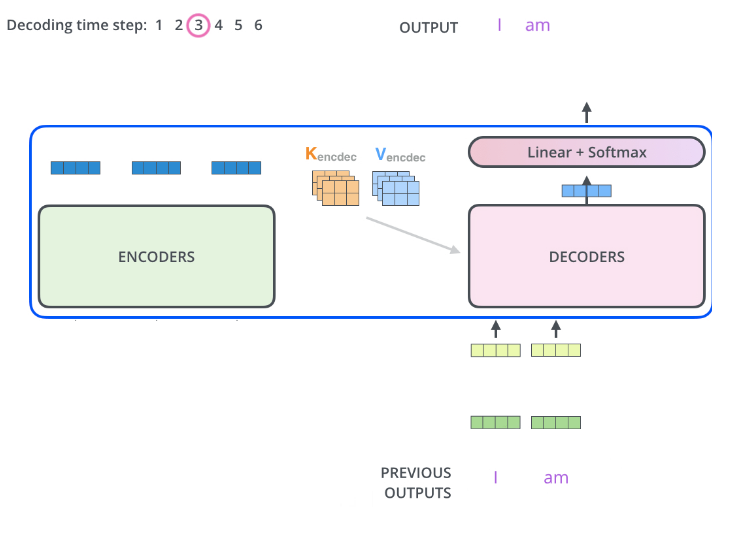
* 1. **Giai đoạn Encoder (tiếp theo)**

Giai đoạn decoder cũng gồm có số lượng lớp xếp chông nhau (stack) bằng với encoder. Kiến ​​trúc decoder cũng tương tự như encoder, nhưng sử dụng thêm lớp bổ sung Mask Multi Head Attention. Lớp bổ sung này giúp ngăn chặn các vị trí tham dự vào các vị trí tiếp theo [31].



***Hình 28:*** *Giai đoạn decoder.* [27]

Đầu ra của bộ mã hóa (top of encoder) sau đó được chuyển thành một tập các vectơ chú ý (attention vector) K và V, vector này được sử dụng cho mỗi lớp decoder.



***Hình 29:*** *Sau khi kết thúc giai đoạn encoder, ma trận K và V được đưa vào giai đoạn decoder. Mỗi bước trong giai đoạn decoder sẽ dự đoán ra 1 từ trong chuỗi đầu ra*. [31]

Đầu ra của mỗi bước được đưa đến decoder dưới cùng trong bước tiếp theo và decoder tạo ra kết quả giải mã giống như các encoder đã làm. Giống như các đầu vào encoder, từ được dự đoán sẽ được nhúng và thêm mã hóa vị trí (positional encoder) vào các đầu vào encoder đó để chỉ ra vị trí của mỗi từ. Lớp cuối cùng là lớp tuyến tính và softmax. Lớp tuyến tính là một mạng nơ ron fully connected neural network, biến vectơ được tạo ra ở các lớp chồng giải mã (stack of decoder), thành một vectơ lớn hơn, được gọi là vectơ logits. Lớp softmax sau đó biến những điểm số đó thành phân bố xác suất (từ 0 đến 1). Xác suất cao nhất được chọn và từ được liên kết với nó được tạo ra. [31]

* 1. **Ưu điểm và nhược điểm của kiến trúc Transformer**

Mô hình Transformer xử lý đầu vào có kich thước thay đổi bằng cách sử dụng nhiều lớp xếp chồng seft-attention (stacks of seft-attention layer), khiến cho kiến trúc này có nhiều ưu điểm: [30]

* Nó không đưa ra giả định nào về các mối quan hệ thời gian / không gian (temporal/spatial relationship) trên dữ liệu. Điều này là lý tưởng để xử lý một tập hợp các đối tượng
* Các đầu ra của lớp có thể được tính song song, thay vì một chuỗi như RNN.
* Các dữ liệu ở khoảng cách xa (distant items) có thể ảnh hưởng đến đầu ra của nhau mà không qua nhiều bước RNN hoặc các lớp chập
* Nó có thể học các phụ thuộc tầm xa. Đây là một thách thức trong nhiều nhiệm vụ trình tự.

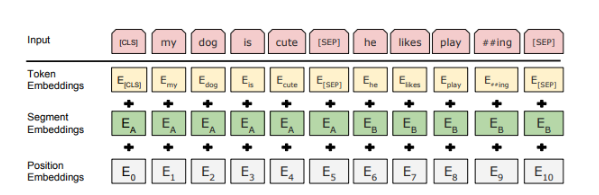
Nhìn về khía cạnh trực quan thì kiến trúc vẫn có những điểm bất lợi:

* Đối với chuỗi thời gian, đầu ra cho một time-step được tính từ toàn bộ lịch sử thay vì chỉ đầu vào và trạng thái ẩn hiện tại. Điều này có thể kém hiệu quả.
* Nếu đầu vào không có mối quan hệ thời gian / không gian, như văn bản, một số mã hóa vị trí (positional encoding) phải được thêm vào.

1. **BERT**

BERT là viết tắt của [Bidirectional Encoder Representations from Transformers](https://arxiv.org/abs/1810.04805) được hiểu là một mô hình học sẵn hay còn gọi là pre-train model, học ra các vector đại diện theo ngữ cảnh 2 chiều của từ, được sử dụng để chuyển sang các bài toán khác trong lĩnh vực xử lý ngôn ngữ tự nhiên [33]. BERT tập trung vào việc cải tiến các phương pháp trong việc xử lý ngôn ngữ tự nhiên bằng việc tìm ra đại diện (representation) của từ thông qua ngữ cảnh của nó trong câu. Vector đại diện được sinh ra từ mô hình BERT được tính chỉnh (fine-turning) với các lớp đầu ra bổ sung đã tạo ra nhiều kiến trúc cải tiến đáng kể cho các nhiệm vụ xử lý ngôn ngữ tự nhiên như Question Answering, Language Inference, v.v... mà không cần thay đổi quá nhiều từ các kiến trúc cũ .[35]

Kiến trúc của mô hình BERT là một kiến trúc đa tầng gồm nhiều lớp **Bidirectional Transformer encoder** dựa trên bản mô tả về mô hình Transformer do [Vaswani](https://arxiv.org/abs/1706.03762) và cộng sự đề xuất [22]. Kiến trúc mô hình BERT đề xuất có 2 nhiệm vụ là pre-training và fine turning, nhiệm vụ của pre-training sẽ là tạo ra vector đại diện theo ngữ cảnh 2 chiều (bidirectional) cho dữ liệu đầu vào, dựa trên giai đoạn encoder của mô hình Transformers với nhiệm vụ tinh chỉnh (fine-turning) đại diện vector này sẽ được ứng dụng vào các nhiệm vụ xử lý ngôn ngữ tự nhiên, tùy theo mục đích ứng dụng.



***Hình 30:*** *Biểu diễn đầu vào pre-training của BERT*

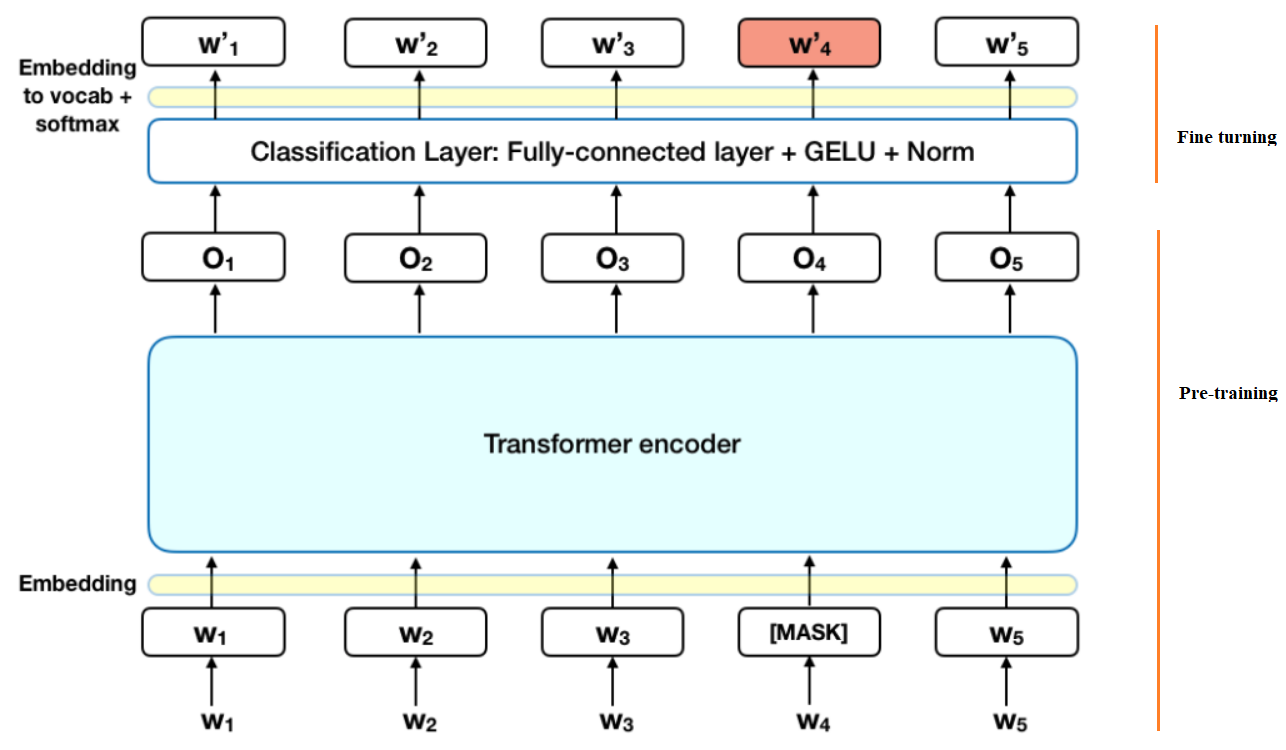
Biểu diễn đầu vào của BERT là một input emdedding gồm có token embeddings, segment embedding và position embeddings.

Giai đoạn Pre-training, có 2 nhiệm vụ là Masked LM và Next Sentence Prediction.

Masked LM sử dụng một cách tiếp cận đơn giản để che giấu đi một số token đầu vào một cách ngẫu nhiên (chọn 15% token) và sau đó chúng ta chỉ dự đoán các token được giấu đi đó. Transformer encoder không hề biết được từ nào sẽ được yêu cầu dự đoán hoặc từ nào đã được thay thế bằng một từ ngẫu nhiên, do đó, nó buộc phải giữ một biểu diễn theo ngữ cảnh của mối token đầu vào. Ngoài ra, do thay thế 1.5% tất cả các tokens bằng một từ ngẫu nhiên nên điều này dường như sẽ không làm ảnh hưởng tới khả năng hiểu ngôn ngữ của mô hình.

Next Sentence Prediction, để đào tạo được mô hình hiểu được mối quan hệ giữa các câu, mô hình cần có thể đoán câu tiếp theo dựa vào câu hiện tại, dữ liệu huẩn luyện có thể là một corpus bất kỳ nào. Khi chọn câu A và câu B cho mỗi training sample, 50% khả năng câu B là câu tiếp theo sau câu A và 50% còn lại là một câu ngẫu nhiên nào đó trong corpus. [35]

Giai đoạn Fine-turning, giai đoạn này tùy theo mục đích sử dụng vector đại diện của dữ liệu đầu vào ở giai đoạn pre-training (output layer) để ứng dụng vào xử lý các nhiệm vụ khác trọng ngôn ngữ tự nhiê như question answering and language inference, without substantial taskspecific architecture modifications.



***Hình 31:*** *Ứng dụng Bert vào bài toán phân loại từ.*

Sau khi qua mô hình encoder Transformer, lớp đầu ra (output layer) O chính là vector đại diện thể hiện ngữ cảnh 2 chiều của dữ liệu đầu vào W. Các vector ở lớp đầu ra (ouput layer) này được đưa vào tiến trình fine-turning sử dụng để giải quyết bài toán phân loại các từ trong câu.

# Chương IV. Hiện thực hóa hệ thống

1. **Dataset**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Bộ dữ liệu** | **Top** | **Bottom** | **Số cặp** | **Số câu bình luận** |
| WoW | 19890 | 15996 | 24417 | - |
| Street2Shop | - | - | 39479 | - |
| Fashion-136K | - | - | 135893 | - |
| FashionVC | 14871 | 13663 | 20726 | - |
| ExpFashion | 29113 | 20902 | 200745 | 1052821 |

- Bộ từ vựng: 92295 từ, thực nghiệm: 16515 từ (tần suất xuất hiện trên 5).

- Số lượng ảnh là 50015 ( 29113 top + 20902 bottom).

- Số lượng cặp ảnh trang phục là 200745 cặp.

- Validate : random 1000 ảnh (top + down)

- Test : random 2000 ảnh (top + down)

1. **Môi trường**

Google Colab Service, ram được cung cấp là 12GB, dung lượng disk là 358.27GB.

Thực hiện huấn luyện mô hình và chạy đánh giá với 5 epochs. Cấu hình máy chủ như trên, thời gian huấn luyện với mỗi epoch tương đương 5 giờ 30 phút với bộ dữ liệu được giới thiệu.

1. **Kết quả**

|  |  |  |
| --- | --- | --- |
| **Cặp trang phục** | | **Câu bình luận** |
| **Top** | **Bottom** |
|  |  | surely you do , and this is sweet like cake ! |
|  |  | hey ! thanks for following us ! |
|  |  | this is darling ! : ) . |

# Chương V. Tổng kết và hướng phát triển

1. **Các công việc đã làm**

- Nghiên cứu các mô hình để xử lý các bài toán xử lý hình ảnh, văn bản trong Deep Learning, Machine Learning.

- Tìm hiểu ứng dụng giải quyết các vấn đề /bài toán về gợi ý và phát sinh câu bình luận cho những bộ trang phục.

- Nghiên cứu mô hình, kỹ thuật để ứng dụng vào việc giải quyết mô hình phát sinh câu bình luận từ ảnh tĩnh.

- Hiện thực lại mô hình “Tự động phát sinh bình luận cho ảnh thời trang”.

1. **Hạn chế**

- Cơ sở vật chất chưa đủ tốt.

- Chưa quen với môi trường làm việc Google Colab Service.

- Chưa có kinh nghiệm khi tiến hành thực nghiệm.

- Kết quả trong việc thực nghiệm chưa thật sự được như mong muốn, nguyên nhân chủ yếu do dataset quá nhiều.

1. **Hướng phát triển**

- Thu thập hoặc tái chỉnh sữa lại toàn bộ dữ liệu huấn luyện

- Thực hiện việc huấn luyện lại mô hình với tập dữ liệu lớn

- Áp dụng thực nghiệm các kỹ thuật như Attribute Recognization, Visual Attention Model,… để phát triển mô hình căn cứ vào những thuộc tính để dựa vào đó đưa ra câu bình luận đáng tin và hợp lý hơn cho người dùng.

# TÀI LIỆU THAM KHẢO

[1] Tomoharu Iwata, Shinji Watanabe and Hiroshi Sawada (2011), “Fashion coordinates recommender system using photographs from fashion magazines“ International Joint Conference on Artificial Intelligence, pp. 2262–2267.

[2] Si Liu, Jiashi Feng, Zheng Song, Tianzhu Zhang, Hanqing Lu, Changsheng Xu and Shuicheng Yan, “Hi, magic closet, tell me what to wear!” in ACM Multimedia, 2012, pp. 619–628.

[3] Yujie Lin (5 Mar 2019), “Explainable Outfit Recommendation with Joint Outfit Matching and Comment Generation”, ” <https://arxiv.org/pdf/1806.08977.pdf>”

[4] Ashish Vaswani et al (6 Dec 2017), “Attention Is All You Need”, “<https://arxiv.org/pdf/1706.03762.pdf>”

[5] Wikipedia, Deep Learning, “<https://en.wikipedia.org/wiki/Deep_learning>”

[6] Wikipedia, A Deep neural network (DNN), “<https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks>”

[7] Ilya Sutskever, Oriol Vinyals, Quoc V. Le (14 Dec 2014), “Sequence to Sequence Learning with Neural Networks”, “<https://arxiv.org/pdf/1409.3215.pdf>”

[8] Encoder-Decoder Architectures ([Mar 7, 2017](https://www.quora.com/What-is-an-Encoder-Decoder-in-Deep-Learning/answer/Christian-Baumgartner-3)), “<https://www.quora.com/What-is-an-Encoder-Decoder-in-Deep-Learning>”

[9] Kyunghyun Cho Bart van Merrienboer Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares Holger Schwenk, Yoshua Bengio (3 Sep 2014), “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”, “<https://arxiv.org/pdf/1406.1078.pdf>”

[10]  [Jason Brownlee](https://machinelearningmastery.com/author/jasonb/) (October 13, 2017), “How Does Attention Work in Encoder-Decoder Recurrent Neural Networks”, “<https://machinelearningmastery.com/how-does-attention-work-in-encoder-decoder-recurrent-neural-networks/>”

[11] Lilian Weng (Jun 24 2018), “Attention? Attention!” , “<https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html#whats-wrong-with-seq2seq-model>”

[12] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, Yoshua Bengio (11 Dec 2014), “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”, “<https://arxiv.org/pdf/1412.3555.pdf>”

[13] Problem of RNN and understanding LSTM Networks (*August 27, 2015)*,

“<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>”

[14] [Ta-Chun Su](https://medium.com/@bgg?source=user_popover) (3 Oct 20118), “Seq2seq pay Attention to Self Attention”, ” <https://medium.com/@bgg/seq2seq-pay-attention-to-self-attention-part-1-d332e85e9aad>”

[15] Dzmitry Bahdanau, KyungHyun Cho và Yoshua Bengio (19 May 2016), “NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE”, “<https://arxiv.org/pdf/1409.0473.pdf>”

[16] Phan Huy Hoang (16 Jan 2017), “ Attention, Attention, Attention,” “<https://viblo.asia/p/machine-learning-attention-attention-attention-eW65GPJYKDO>”

[17] Alex Graves et…al (10 Dec 2014), “Neural Turing Machines”, “<https://arxiv.org/pdf/1410.5401.pdf>”

[18] Minh-Thang Luong et..al (20 Sep 2015), “Effective Approaches to Attention-based Neural Machine Translation”, “<https://arxiv.org/pdf/1508.04025.pdf>”

[19] Ashish Vaswani and Anna Huang (2016), “Self-Attention For Generative Models”, “<https://web.stanford.edu/class/cs224n/slides/cs224n-2019-lecture14-transformers.pdf>”

[20] Aditya Thiruvengadam (Oct 9, 2018), “Transformer Architecture: Attention Is All You Need” , “<https://medium.com/@adityathiruvengadam/transformer-architecture-attention-is-all-you-need-aeccd9f50d09>”

[21] Hojjat Salehinejad et..al (22 Feb 2018), “Recent Advances in Recurrent Neural Networks, <https://arxiv.org/pdf/1801.01078.pdf>

[22] Ashish Vaswani et…al (6 Dec 2017), “Attention Is All You Need”, “<https://arxiv.org/pdf/1706.03762.pdf>”

[23] [Keitakurita](http://mlexplained.com/author/admin/) (29 Dec 2017 ), “Explanin Attention Is All You Need”, <http://mlexplained.com/2017/12/29/attention-is-all-you-need-explained/>

[24] Gongbo Tang1 et…al (11 Nov 2018), “Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures”, “<https://arxiv.org/pdf/1808.08946.pdf>”

[25] [Alibaba Cloud](https://medium.com/@Alibaba_Cloud) (12 Sep 2018 ), “ Self-Attention Mechanisms in Natural Language Processing”, ” <https://medium.com/@Alibaba_Cloud/self-attention-mechanisms-in-natural-language-processing-9f28315ff905>”

[26] Zhixing Tan , Mingxuan Wang et…al (5 Dec 2017), ”Deep Semantic Role Labeling with Self-Attention”, “<https://arxiv.org/pdf/1712.01586.pdf>”

[27] (Intra-att)Ankur P. Parikh et…al (2016), “A Decomposable Attention Model for Natural Language Inference”, <https://aclweb.org/anthology/D16-1244>

[28] Peter Shaw et…al (2017), Self-Attention with Relative Position Representations, <https://arxiv.org/pdf/1803.02155.pdf>

[29] [Maxime Allard](https://medium.com/@maxime.allard) (Jan 5 2019), “What is a Transformer?“, “<https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04>“

[30] Tensorflow, “Transformer model for language understanding”, “<https://www.tensorflow.org/beta/tutorials/text/transformer>”

[31] Jay Alammar (June 27 2018,) “The Illustrated Transformer”, “<http://jalammar.github.io/illustrated-transformer/>“

[32] Giuliano Giacaglia (Mar 11 2018), “How Transformers Work”, “<https://towardsdatascience.com/transformers-141e32e69591>”

[33] Michal Chromiak (Tue, 12 Sep 2017) “The Transformer – Attention is all you need”, “<https://mchromiak.github.io/articles/2017/Sep/12/Transformer-Attention-is-all-you-need/#positional-encoding-pe>”

[34] Jacob Devlin, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding” 24 May 2019, “<https://arxiv.org/pdf/1810.04805.pdf>”

[35] (Sep 30 2018) “Understanding of RECURRENT NEURAL NETWORKS (LSTM, GRU)”, “<https://mc.ai/understanding-of-recurrent-neural-networks-lstm-gru/>”

## [36] SuperDataScience Team (Aug 23 2018) “Recurrent Neural Networks (RNN) - The Vanishing Gradient Problem”, “<https://www.superdatascience.com/blogs/recurrent-neural-networks-rnn-the-vanishing-gradient-problem>”

## [37] [Simeon Kostadinov](https://towardsdatascience.com/@simonnoff) (Dec 16 2017), “Understanding GRU Networks”, ”<https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be>”