Sự chú ý là tất cả những gì bạn cần

Ashish Vaswani
Google Brain
avaswani@google.com

Noam Shazeer
Google Brain
noam@google.com

Niki Parmar
Google Nghiên cứu
nikip@google.com

Jakob Uszkoreit
Google Nghiên cứu
usz@google.com

Llion Jones
Google Nghiên cứu
llion@google.com

Aidan N. Gomez †
Đại học Toronto
aidan@cs.toronto.edu

Łukasz Kaiser
Google Brain
lukaszkaiser@google.com

Illia Polosukhin
‡ illia.polosukhin@gmail.com

trừu tượng

Các mô hình truyền tải trình tự chiếm ưu thế dựa trên các mạng thần kinh tái phát hoặc tích chập phức tạp bao gồm bộ mã hóa và bộ giải mã. Các mô hình hoạt động tốt nhất cũng kết nối bộ mã hóa và bộ giải mã thông qua cơ chế chú ý. Chúng tôi đề xuất một kiến trúc mạng đơn giản mới, Transformer, chỉ dựa trên các cơ chế chú ý, loại bỏ hoàn toàn sự lặp lại và tích chập. Các thử nghiệm trên hai tác vụ dịch máy cho thấy các mô hình này có chất lượng vượt trội, đồng thời có khả năng song song hóa cao hơn và cần ít thời gian đào tạo hơn đáng kể. Mô hình của chúng tôi đạt được 28,4 BLEU trong nhiệm vụ dịch thuật từ tiếng Anh sang tiếng Đức của WMT 2014, cải thiện hơn 2 BLEU so với kết quả tốt nhất hiện có, bao gồm cả các bản hòa tấu. Trong nhiệm vụ dịch từ tiếng Anh sang tiếng Pháp của WMT 2014, mô hình của chúng tôi thiết lập điểm BLEU hiện đại nhất cho một mô hình mới là 41,0 sau khi đào tạo trong 3,5 ngày trên tám GPU, một phần nhỏ chi phí đào tạo của những GPU tốt nhất những mô hình từ văn học

1. Giới thiệu

Mạng thần kinh tái phát, bộ nhớ ngắn hạn dài [12] và mạng thần kinh tái phát có kiểm soát [7] nói riêng, đã được thiết lập vững chắc như là các phương pháp tiếp cận hiện đại trong các vấn đề mô hình hóa và chuyển đổi trình tự như mô hình ngôn ngữ và dịch máy [29, 2 , 5]. Kể từ đó, nhiều nỗ lực đã tiếp tục vượt qua ranh giới của các mô hình ngôn ngữ lặp lại và kiến trúc bộ mã hóa-giải mã [31, 21, 13].

Đóng góp bình đẳng. Thứ tự liệt kê là ngẫu nhiên. Jakob đề xuất thay thế RNN bằng sự tự chú ý và bắt đầu nỗ lực đánh giá ý tưởng này. Ashish, cùng với Illia, đã thiết kế và triển khai các mẫu Transformer đầu tiên và đã tham gia chủ yếu vào mọi khía cạnh của công việc này. Noam đề xuất sự chú ý theo sản phẩm chấm theo tỷ lệ, sự chú ý nhiều đầu và cách biểu diễn vị trí không có tham số và trở thành người khác tham gia vào hầu hết mọi chi tiết. Niki đã thiết kế, triển khai, điều chỉnh và đánh giá vô số biến thể mô hình trong cơ sở mã và tensor2tensor ban đầu của chúng tôi. Llion cũng đã thử nghiệm các biến thể mô hình mới, chịu trách nhiệm về cơ sở mã ban đầu của chúng tôi cũng như khả năng suy luận và trực quan hóa hiệu quả. Lukasz và Aidan đã dành vô số ngày dài để thiết kế các bộ phận khác nhau và triển khai tensor2tensor, thay thế cơ sở mã trước đó của chúng tôi, cải thiện đáng kể kết quả và tăng tốc đáng kể nghiên cứu của chúng tôi.

†Công việc được thực hiện khi làm việc tại Google Brain. ‡Công việc được thực hiện khi làm việc tại Google Research.

Hội nghị lần thứ 31 về Hệ thống xử lý thông tin thần kinh (NIPS 2017), Long Beach, CA, Hoa Kỳ.

Các mô hình lặp lại thường tính toán nhân tử dọc theo vị trí ký hiệu của chuỗi đầu vào và đầu ra. Căn chỉnh các vị trí theo các bước trong thời gian tính toán, chúng tạo ra một chuỗi các trạng thái ẩn ht, như một hàm của trạng thái ẩn trước đó ht 1 và đầu vào cho vị trí t. Bản chất tuần tự vốn có này ngăn cản việc song song hóa trong các ví dụ huấn luyện, điều này trở nên quan trọng ở độ dài chuỗi dài hơn, vì các hạn chế về bộ nhớ hạn chế việc phân nhóm giữa các ví dụ. Công việc gần đây đã đạt được những cải tiến đáng kể về hiệu quả tính toán thông qua các thủ thuật phân tích nhân tử [18] và tính toán có điều kiện [26], đồng thời cải thiện hiệu suất mô hình trong trường hợp sau. Tuy nhiên, hạn chế cơ bản của tính toán tuần tự vẫn còn.

Các cơ chế chú ý đã trở thành một phần không thể thiếu của mô hình chuyển đổi và mô hình trình tự hấp dẫn trong các nhiệm vụ khác nhau, cho phép mô hình hóa các phụ thuộc mà không quan tâm đến khoảng cách của chúng trong trình tự đầu vào hoặc đầu ra [2, 16]. Tuy nhiên, trong tất cả trừ một số trường hợp [22], các cơ chế chú ý như vậy được sử dụng cùng với mạng định kỳ.

Trong công việc này, chúng tôi đề xuất Transformer, một kiến trúc mô hình tránh sự lặp lại và thay vào đó dựa hoàn toàn vào cơ chế chú ý để rút ra sự phụ thuộc tổng thể giữa đầu vào và đầu ra.
Transformer cho phép thực hiện song song nhiều hơn đáng kể và có thể đạt đến trạng thái hiện đại mới về chất lượng dịch thuật sau khi được đào tạo chỉ trong 12 giờ trên tám GPU P100.

2 Bối cảnh

Mục tiêu giảm tính toán tuần tự cũng tạo thành nền tảng của GPU thần kinh mở rộng [20], ByteNet [15] và ConvS2S [8], tất cả đều sử dụng mạng thần kinh tích chập làm khối xây dựng cơ bản, tính toán các biểu diễn ẩn song song cho tất cả đầu vào và các vị trí đầu ra. Trong các mô hình này, số lượng thao tác cần thiết để liên kết các tín hiệu từ hai vị trí đầu vào hoặc đầu ra tùy ý tăng theo khoảng cách giữa các vị trí, tuyến tính đối với ConvS2S và logarit đối với ByteNet. Điều này làm cho việc học sự phụ thuộc giữa các vị trí ở xa trở nên khó khăn hơn [11]. Trong Transformer, điều này được giảm xuống thành một số thao tác không đổi, mặc dù phải trả giá bằng độ phân giải hiệu quả bị giảm do lấy trung bình các vị trí có trọng số chú ý, một hiệu ứng mà chúng tôi chống lại bằng Chú ý nhiều đầu như được mô tả trong phần 3.2.

Tự chú ý, đôi khi được gọi là chú ý nội tâm, là một cơ chế chú ý liên quan đến các vị trí khác nhau của một chuỗi để tính toán cách trình bày chuỗi đó. Khả năng tự chú ý đã được sử dụng thành công trong nhiều nhiệm vụ khác nhau bao gồm đọc hiểu, tóm tắt trừu tượng, rút ra văn bản và học cách trình bày câu độc lập với nhiệm vụ [4, 22, 23, 19].

Mạng bộ nhớ đầu cuối dựa trên cơ chế chú ý lặp lại thay vì lặp lại theo trình tự và đã được chứng minh là hoạt động tốt trong các nhiệm vụ trả lời câu hỏi bằng ngôn ngữ đơn giản và mô hình hóa ngôn ngữ [28].

Tuy nhiên, theo hiểu biết tốt nhất của chúng tôi, Transformer là mô hình tải nạp đầu tiên hoàn toàn dựa vào khả năng tự chú ý để tính toán các biểu diễn đầu vào và đầu ra của nó mà không sử dụng RNN hoặc tích chập được căn chỉnh theo trình tự. Trong các phần sau, chúng tôi sẽ mô tả Máy biến áp, thúc đẩy sự chú ý của bản thân và thảo luận về những ưu điểm của nó so với các mô hình như [14, 15] và [8].

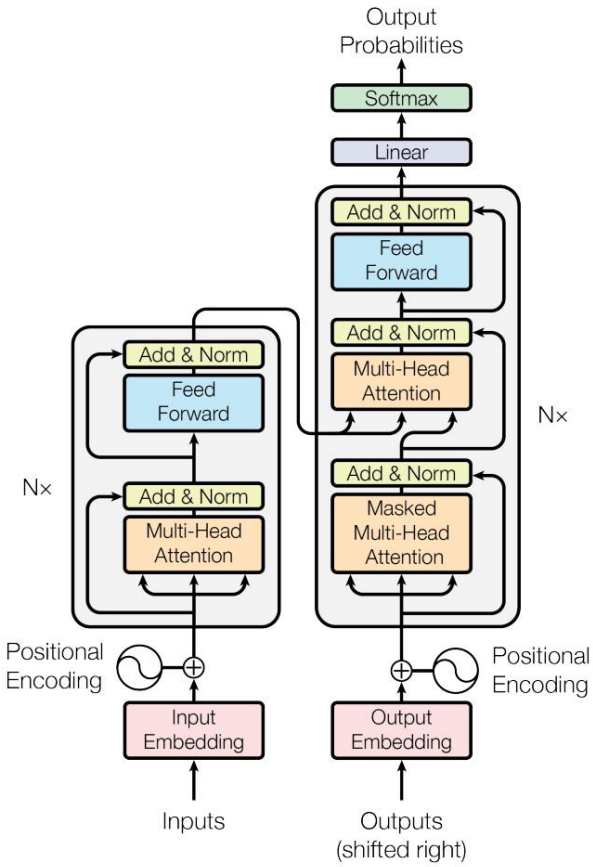
3 Kiến trúc mô hình

Hầu hết các mô hình truyền tải chuỗi thần kinh cạnh tranh đều có cấu trúc bộ mã hóa-giải mã [5, 2, 29]. Ở đây, bộ mã hóa ánh xạ chuỗi đầu vào của các biểu diễn ký hiệu (x1, ..., xn) thành một chuỗi các biểu diễn liên tục z = (z1, ..., zn). Cho z, bộ giải mã sau đó tạo ra một chuỗi đầu ra (y1, ..., ym) gồm các ký hiệu, mỗi phần tử một. Ở mỗi bước, mô hình sẽ tự động hồi quy [9], sử dụng các ký hiệu được tạo trước đó làm đầu vào bổ sung khi tạo ký hiệu tiếp theo.

Transformer tuân theo kiến trúc tổng thể này bằng cách sử dụng các lớp tự chú ý và điểm thông minh xếp chồng lên nhau, được kết nối đầy đủ cho cả bộ mã hóa và bộ giải mã, tương ứng được hiển thị ở nửa bên trái và bên phải của Hình 1 .

3.1 Ngăn xếp bộ mã hóa và giải mã

Bộ mã hóa: Bộ mã hóa bao gồm một chồng N = 6 lớp giống hệt nhau. Mỗi lớp có hai lớp con. Đầu tiên là cơ chế tự chú ý nhiều đầu, và thứ hai là cơ chế đơn giản, định vị-



Hình 1: Máy biến áp - kiến trúc mô hình.

mạng chuyển tiếp nguồn cấp dữ liệu được kết nối đầy đủ khôn ngoan. Chúng tôi sử dụng kết nối dư [10] xung quanh mỗi lớp trong số hai lớp con, sau đó là chuẩn hóa lớp [1]. Nghĩa là, đầu ra của mỗi lớp con là LayerNorm(x + Sublayer(x)), trong đó Sublayer(x) là chức năng do chính lớp con đó thực hiện . Để tạo điều kiện thuận lợi cho các kết nối còn lại này, tất cả các lớp con trong mô hình cũng như các lớp nhúng tạo ra kết quả đầu ra có kích thước dmodel = 512.

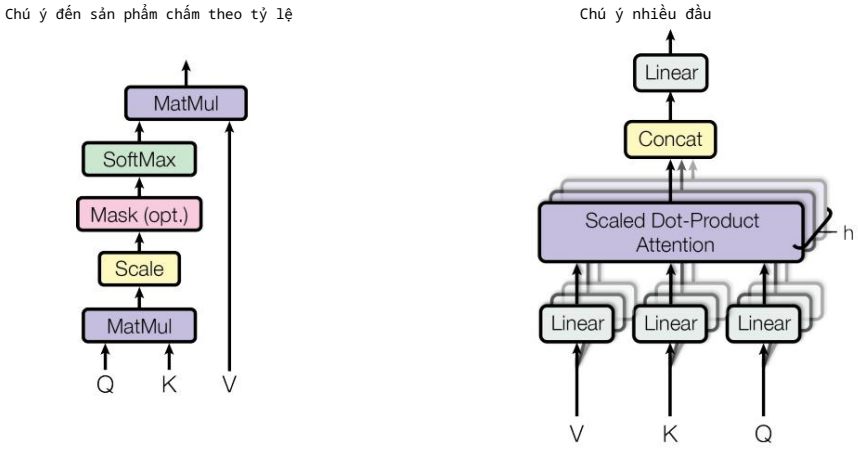
Bộ giải mã: Bộ giải mã cũng bao gồm một chồng gồm N = 6 lớp giống hệt nhau. Ngoài hai lớp con trong mỗi lớp bộ mã hóa, bộ giải mã còn chèn một lớp con thứ ba, lớp này thực hiện sự chú ý nhiều đầu đối với đầu ra của ngăn xếp bộ mã hóa. Tương tự như bộ mã hóa, chúng tôi sử dụng các kết nối còn lại xung quanh mỗi lớp con, sau đó là chuẩn hóa lớp. Chúng tôi cũng sửa đổi lớp con tự chú ý trong ngăn xếp bộ giải mã để ngăn các vị trí tham gia vào các vị trí tiếp theo. Việc che giấu này , kết hợp với thực tế là các phần nhúng đầu ra được bù bởi một vị trí, đảm bảo rằng các dự đoán cho vị trí i chỉ có thể phụ thuộc vào các đầu ra đã biết ở các vị trí nhỏ hơn i.

3.2 Chú ý

Hàm chú ý có thể được mô tả như ánh xạ một truy vấn và một tập hợp các cặp khóa-giá trị tới đầu ra, trong đó truy vấn, khóa, giá trị và đầu ra đều là vectơ. Đầu ra được tính dưới dạng tổng có trọng số của các giá trị, trong đó trọng số được gán cho mỗi giá trị được tính bằng hàm tương thích của truy vấn với khóa tương ứng.

3.2.1 Chú ý đến sản phẩm chấm theo tỷ lệ

Chúng tôi gọi sự chú ý đặc biệt của mình là "Sự chú ý đến sản phẩm chấm theo tỷ lệ" (Hình 2). Đầu vào bao gồm các truy vấn và khóa có kích thước dk và các giá trị của kích thước dv. Chúng tôi tính toán các sản phẩm chấm của



Hình 2: (trái) Chú ý đến sản phẩm chấm theo tỷ lệ. (phải) Chú ý nhiều đầu bao gồm một số lớp chú ý chạy song song.

truy vấn bằng tất cả các khóa, chia từng khóa cho $\sqrt{d_k}$ và áp dụng hàm softmax để thu được trọng số trên các giá trị.

Trong thực tế, chúng tôi tính toán đồng thời hàm chú ý trên một tập hợp truy vấn, được đóng gói cùng nhau thành ma trận Q. Các khóa và giá trị cũng được đóng gói cùng nhau thành ma trận K và V . Chúng tôi tính toán ma trận đầu ra như sau:

$$\text{Chú ý}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \quad (1)$$

Hai hàm chú ý được sử dụng phổ biến nhất là chú ý cộng [2] và chú ý tích số chấm (nhân). Sự chú ý của sản phẩm chấm giống hệt với thuật toán của chúng tôi, ngoại trừ hệ số tỷ lệ là $\frac{1}{\sqrt{d_k}}$. Sự chú ý bổ sung tính toán chức năng tương thích bằng cách sử dụng mạng chuyển tiếp nguồn cấp dữ liệu với một lớp ẩn duy nhất. Mặc dù cả hai đều giống nhau về độ phức tạp về mặt lý thuyết, như ng sự chú ý của tích số chấm nhanh hơn và tiết kiệm không gian hơn trong thực tế vì nó có thể được thực hiện bằng cách sử dụng mã nhân ma trận được tối ưu hóa cao.

Trong khi đối với các giá trị nhỏ của d_k thì hai cơ chế hoạt động tương tự nhau, sự chú ý cộng dồn vượt trội hơn sự chú ý của sản phẩm chấm mà không cần chia tỷ lệ đối với các giá trị lớn hơn của d_k [3]. Chúng tôi nghi ngờ rằng đối với các giá trị lớn của d_k, tích số chấm tăng theo độ lớn, đẩy hàm softmax vào các vùng có độ dốc cực nhỏ [4]. Để chống lại hiệu ứng này, chúng tôi chia tỷ lệ tích số chấm theo $\frac{1}{\sqrt{d_k}}$.

3.2.2 Sự chú ý của nhiều đầu

Thay vì thực hiện một chức năng chú ý duy nhất với các khóa, giá trị và truy vấn theo chiều dmodel, chúng tôi nhận thấy việc chiếu tuyến tính các truy vấn, khóa và giá trị h lần với các phép chiếu tuyến tính đã học khác nhau cho các kích thước dk, dk và dv tương ứng sẽ có ích. Sau đó , trên mỗi phiên bản truy vấn, khóa và giá trị dự kiến này, chúng tôi sẽ thực hiện song song chức năng chú ý, mang lại giá trị đầu ra theo chiều dv . Chúng được nối với nhau và chiếu lại một lần nữa, dẫn đến các giá trị cuối cùng, như được mô tả trong Hình 2.

Sự chú ý của nhiều người cho phép mô hình cùng tham gia vào thông tin từ các không gian con biểu diễn khác nhau ở các vị trí khác nhau. Với một đầu chú ý duy nhất, tính trung bình sẽ hạn chế điều này.

4Để minh họa tại sao tích số chấm lớn hơn, giả sử rằng các thành phần của q và k là qiki ngẫu nhiên độc lập, các biến có trung bình 0 và phương sai 1. Khi đó tích vô hướng của chúng, $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$ có trung bình 0 và phương sai d_k.

MultiHead(Q, K, V) = Concat(head1, ..., headh)WO trong đó

headi = Chú ý(QWQ tôi ,KWK tôi , V WV tôi)

Trong đó các phép chiếu là ma trận tham số W và WO Q i ∈ R dmodel×dk , Wi K ∈ R dmodel×dk , WV i ∈ R dmodel×dv và WO ∈ R hdv×dmodel .

Trong công việc này, chúng tôi sử dụng h = 8 lớp chú ý song song hoặc các đầu. Đối với mỗi cái này, chúng tôi sử dụng dk = dv = dmodel/h = 64. Do kích thước của mỗi đầu giảm đi, tổng chi phí tính toán tương tự như chi phí tính toán của sự chú ý một đầu với đầy đủ chiều.

3.2.3 Ứng dụng của Sự chú ý trong Mô hình của chúng tôi

Transformer sử dụng sự chú ý của nhiều đầu theo ba cách khác nhau:

- Trong lớp "chú ý bộ mã hóa-bộ giải mã", các truy vấn đến từ lớp bộ giải mã trước đó, còn các khóa và giá trị bộ nhớ đến từ đầu ra của bộ mã hóa. Điều này cho phép mọi vị trí trong bộ giải mã tham dự trên tất cả các vị trí trong chuỗi đầu vào. Điều này bắt chước cơ chế chú ý của bộ mã hóa-giải mã điển hình trong các mô hình tuần tự như [31, 2, 8].

- Bộ mã hóa chứa các lớp tự chú ý. Trong lớp tự chú ý, tất cả các khóa, giá trị và truy vấn đều đến từ cùng một nơi, trong trường hợp này là đầu ra của lớp trước đó trong bộ mã hóa. Mỗi vị trí trong bộ mã hóa có thể tham dự tất cả các vị trí ở lớp trước của bộ mã hóa.

- Tương tự, các lớp tự chú ý trong bộ giải mã cho phép mỗi vị trí trong bộ giải mã tham gia vào tất cả các vị trí trong bộ giải mã cho đến và bao gồm cả vị trí đó. Chúng ta cần ngăn luồng thông tin sang trái trong bộ giải mã để bảo toàn đặc tính tự hồi quy. Chúng tôi triển khai điều này bên trong sự chú ý của tích số chấm được chia tỷ lệ bằng cách che đi (đặt thành −∞) tất cả các giá trị trong đầu vào của softmax tương ứng với các kết nối bất hợp pháp. Xem Hình 2.

3.3 Mạng chuyển tiếp nguồn cấp dữ liệu theo vị trí

Ngoài các lớp con chú ý, mỗi lớp trong bộ mã hóa và bộ giải mã của chúng tôi còn chứa một mạng chuyển tiếp nguồn cấp dữ liệu được kết nối đầy đủ, được áp dụng cho từng vị trí riêng biệt và giống hệt nhau. Điều này bao gồm hai phép biến đổi tuyến tính với sự kích hoạt ReLU ở giữa.

$$FFN(x) = \max(0, xW1 + b1)W2 + b2 \quad (2)$$

Mặc dù các phép biến đổi tuyến tính giống nhau ở các vị trí khác nhau nhưng chúng sử dụng các tham số khác nhau giữa các lớp. Một cách khác để mô tả điều này là hai tích chập có kích thước hạt nhân là 1.
Chiều của đầu vào và đầu ra là dmodel = 512 và lớp bên trong có chiều df f = 2048.

3.4 Nhúng và Softmax

Tương tự như các mô hình chuyển đổi trình tự khác, chúng tôi sử dụng các phần nhúng đã học để chuyển đổi mã thông báo đầu vào và mã thông báo đầu ra thành vectơ của mô hình thứ nguyên. Chúng tôi cũng sử dụng phép biến đổi tuyến tính đã học và hàm softmax thông thường để chuyển đổi đầu ra bộ giải mã thành xác suất mã thông báo tiếp theo được dự đoán. Trong mô hình của chúng tôi, chúng tôi chia sẻ cùng một ma trận trọng số giữa hai lớp nhúng và phép biến đổi tuyến tính tiền softmax, tương tự như [24]. Trong các lớp nhúng, chúng tôi nhân các trọng số đó với √ dmodel.

3.5 Mã hóa vị trí

Vì mô hình của chúng tôi không có phép lặp và không tích chập nên để mô hình sử dụng thứ tự của chuỗi, chúng tôi phải đưa vào một số thông tin về vị trí tương đối hoặc tuyệt đối của các mã thông báo trong chuỗi. Để đạt được mục đích này, chúng tôi thêm "mã hóa vị trí" vào phần nhúng đầu vào tại

Bảng 1: Độ dài đường dẫn tối đa, độ phức tạp trên mỗi lớp và số lượng hoạt động tuần tự tối thiểu cho các loại lớp khác nhau. n là độ dài chuỗi, d là kích thước biểu diễn, k là hạt nhân kích thước của các kết cấu và r kích thước của vùng lân cận trong khả năng tự chú ý bị hạn chế.

Loại lớp	Độ phức tạp trên mỗi lớp	Hoạt động	Độ dài đường dẫn tối đa tuần tự
Tự chú ý	TRÊ N^2 ·	O(1)	O(1)
Định kỳ	O(n · d^2 d))	TRÊ N)	TRÊ N)
tích chập	O(k · n · d^2)	O(1)	O(logk(n))
Tự chú ý (hạn chế)	O(r · n · d)	O(1)	O(n/r)

đáy của ngăn xếp bộ mã hóa và bộ giải mã. Các mã hóa vị trí có cùng chiều dmodel như các phần nhúng, để cả hai có thể được tóm tắt. Có nhiều lựa chọn về mã hóa vị trí, đã học và sửa lỗi [8].

Trong công việc này, chúng tôi sử dụng các hàm sin và cosin có tần số khác nhau:

$$P E(pos,2i) = \sin(pos/100002i/dmodel)$$

$$P E(pos,2i+1) = \cos(pos/100002i/dmodel)$$

trong đó pos là vị trí và i là kích thước. Nghĩa là, mỗi chiều của mã hóa vị trí tương ứng với một hình sin. Các bước sóng tạo thành một cấp số nhân từ 2π đến 10000 · 2π. Chúng tôi đã chọn chức năng này vì chúng tôi đưa ra giả thuyết rằng nó sẽ cho phép mô hình dễ dàng học cách tham dự các vị trí tương đối, vì đối với bất kỳ độ lệch k cố định nào, P Epos+k có thể được biểu diễn dưới dạng hàm tuyến tính của P Epos.

Thay vào đó , chúng tôi cũng đã thử nghiệm bằng cách sử dụng các phần nhúng vị trí đã học [8] và nhận thấy rằng cả hai các phiên bản tạo ra kết quả gần như giống hệt nhau (xem hàng Bảng 3 (E)). Chúng tôi chọn phiên bản hình sin bởi vì nó có thể cho phép mô hình ngoại suy theo độ dài chuỗi dài hơn độ dài chuỗi gặp phải Trong quá trình huấn luyện.

4 Tại sao phải chú ý đến bản thân

Trong phần này, chúng tôi so sánh các khía cạnh khác nhau của các lớp tự chú ý với các lớp lặp lại và lớp chập thường được sử dụng để ánh xạ một chuỗi biểu diễn ký hiệu có độ dài thay đổi (x1, ..., xn) sang một dãy khác có độ dài bằng nhau (z1, ..., zn), với xi ,zi R ^{d}, chẳng hạn như một ẩn lớp trong bộ mã hóa hoặc bộ giải mã truyền dẫn trình tự điển hình. Thúc đẩy việc sử dụng sự chú ý của chúng ta hãy xem xét ba mong muốn.

Một là tổng độ phức tạp tính toán trên mỗi lớp. Một điều nữa là số lượng tính toán có thể được song song hóa, được đo bằng số lượng hoạt động tuần tự tối thiểu được yêu cầu.

Thứ ba là độ dài đường dẫn giữa các phần phụ thuộc tầm xa trong mạng. Học tập lâu dài sự phụ thuộc là một thách thức chính trong nhiều nhiệm vụ chuyển đổi trình tự. Một yếu tố quan trọng ảnh hưởng đến khả năng tìm hiểu các phụ thuộc như vậy là độ dài của các đường dẫn tín hiệu tiến và lùi phải có truyền trong mạng. Các đường dẫn này giữa bất kỳ tổ hợp vị trí nào trong đầu vào càng ngắn và trình tự đầu ra thì việc học các phụ thuộc tầm xa càng dễ dàng [11]. Do đó chúng tôi cũng so sánh độ dài đường dẫn tối đa giữa hai vị trí đầu vào và đầu ra bất kỳ trong các mạng bao gồm các loại lớp khác nhau.

Như đã lưu ý trong Bảng 1, lớp tự chú ý kết nối tất cả các vị trí với số lượng liên tục không đổi các hoạt động được thực hiện, trong khi lớp lặp lại yêu cầu các hoạt động tuần tự O(n) . Về mặt độ phức tạp tính toán, các lớp tự chú ý sẽ nhanh hơn các lớp lặp lại khi chuỗi chiều dài n nhỏ hơn chiều biểu diễn d, điều này thường xảy ra với cách trình bày câu được sử dụng bởi các mô hình tiên tiến nhất trong các bản dịch máy, chẳng hạn như từng từ [31] và biểu diễn cặp byte [25] . Để cải thiện hiệu suất tính toán cho các nhiệm vụ liên quan đến các chuỗi rất dài, sự tự chú ý có thể bị hạn chế khi chỉ xem xét một vùng lân cận có kích thước r trong

trình tự đầu vào tập trung quanh vị trí đầu ra tương ứng. Điều này sẽ tăng độ dài đường dẫn tối đa lên O(n/r). Chúng tôi dự định điều tra phương pháp này hơn nữa trong công việc trong tương lai.

Một lớp chập đơn có độ rộng hạt nhân k < n không kết nối tất cả các cặp vị trí đầu vào và đầu ra. Làm như vậy đòi hỏi một chồng các lớp chập O(n/k) trong trường hợp các hạt nhân liền kề hoặc O(logk(n)) trong trường hợp các lớp chập bị giãn [15], tăng độ dài của các đường đi dài nhất giữa hai vị trí bất kỳ trong mạng. Các lớp tích chập thường đắt hơn các lớp hồi quy, theo hệ số k. Tuy nhiên, các tích chập có thể tách rời [6] làm giảm độ phức tạp đáng kể, đến O(k · n · d + n · d tích chập tương đương với sự kết hợp giữa lớp tự chú ý và lớp chuyển tiếp theo điểm, cách tiếp cận mà chúng tôi tham gia ^{2}). Tuy nhiên, ngay cả với k = n, độ phức tạp của một hàm phân tách được vào mô hình của chúng tôi.

Về lợi ích phụ, việc tự chú ý có thể mang lại những mô hình dễ hiểu hơn. Chúng tôi kiểm tra sự phân bổ sự chú ý từ các mô hình của mình, đồng thời trình bày và thảo luận các ví dụ trong phần phụ lục. Những người chú ý riêng lẻ không chỉ học cách thực hiện các nhiệm vụ khác nhau một cách rõ ràng mà nhiều người còn thể hiện hành vi liên quan đến cấu trúc cú pháp và ngữ nghĩa của câu.

5 Đào tạo

Phần này mô tả chế độ đào tạo cho các mô hình của chúng tôi.

5.1 Dữ liệu huấn luyện và phân khối

Chúng tôi đã đào tạo trên bộ dữ liệu Anh-Đức tiêu chuẩn WMT 2014 bao gồm khoảng 4,5 triệu cặp câu. Các câu được mã hóa bằng cách sử dụng mã hóa cặp byte [3], có vốn từ vựng nguồn-đích dùng chung khoảng 37000 mã thông báo. Đối với tiếng Anh-Pháp, chúng tôi đã sử dụng bộ dữ liệu tiếng Anh-Pháp WMT 2014 lớn hơn đáng kể bao gồm 36 triệu câu và chia mã thông báo thành một từ vựng gồm 32000 từ [31]. Các cặp câu được nhóm lại với nhau theo độ dài chuỗi gần đúng. Mỗi đợt huấn luyện chứa một tập hợp các cặp câu chứa khoảng 25000 mã thông báo nguồn và 25000 mã thông báo đích.

5.2 Phần cứng và lịch trình

Chúng tôi đã đào tạo các mô hình của mình trên một máy có 8 GPU NVIDIA P100. Đối với các mô hình cơ sở của chúng tôi sử dụng siêu tham số được mô tả trong suốt bài viết, mỗi bước huấn luyện mất khoảng 0,4 giây. Chúng tôi đã huấn luyện các mô hình cơ sở với tổng số 100.000 bước hoặc 12 giờ. Đối với các mô hình lớn của chúng tôi (được mô tả ở dòng cuối cùng của bảng 3), thời gian bước là 1,0 giây. Những người mẫu lớn được huấn luyện 300.000 bước (3,5 ngày).

5.3 Trình tối ưu hóa

Chúng tôi đã sử dụng trình tối ưu hóa Adam [17] với β1 = 0,9, β2 = 0,98 và = 10 9 . Chúng tôi thay đổi tốc độ học tập trong suốt quá trình đào tạo, theo công thức:

$$\text{hình} \quad 0{,}5 \; \underset{\text{lrate} = d}{\overset{\text{mô}}{\cdot}} \; \text{phút}(\text{bước\_num} \quad 0{,}5 \; , \; \text{bước\_num} \cdot \text{bước khởi động} \quad 1{,}5 \;) \tag{3}$$

Điều này tương ứng với việc tăng tốc độ học tập một cách tuyến tính cho các bước huấn luyện Warmup_steps đầu tiên và giảm tốc độ học tập sau đó tỷ lệ thuận với căn bậc hai nghịch đảo của số bước. Chúng tôi đã sử dụng Warmup_steps = 4000.

5.4 Chính quy hóa

Chúng tôi sử dụng ba loại chính quy trong quá trình đào tạo:

Residual Dropout Chúng tôi áp dụng dropout [27] cho đầu ra của mỗi lớp con, trước khi nó được thêm vào đầu vào của lớp con và được chuẩn hóa. Ngoài ra, chúng tôi áp dụng loại bỏ đối với tổng của các phần nhúng và mã hóa vị trí trong cả ngăn xếp bộ mã hóa và bộ giải mã. Đối với mô hình cơ sở, chúng tôi sử dụng tỷ lệ Pdrop = 0,1.

Bảng 2: Máy biến áp đạt đư ợc điểm BLEU tốt hơn so với các mẫu máy tiên tiến nhất trư ớc đây trên
Các bài kiểm tra tin tức từ tiếng Anh sang tiếng Đức và tiếng Anh sang tiếng Pháp năm 2014 với chi phí đào tạo chỉ bằng một phần nhỏ.

Ngư ời mẫu	BLEU		Chi phí đào tạo (FLOP)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [15]	23,75			
Deep-Att + PosUnk [32]		39,2		1,0 · 1020
GNMT + RL [31]	24,6	39,92	2,3 · 1019	1,4 · 1020
Chuyển đổiS2S [8]	25,16	40,46	9,6 · 1018	1,5 · 1020
Bộ GD [26]	26,03	40,56	2,0 · 1019	1,2 · 1020
Bộ đồng phục Deep-Att + PosUnk [32]		40,4		8,0 · 1020
Bộ hòa tấu GNMT + RL [31]	26:30	41,16	1,8 · 1020	1,1 · 1021
Nhóm ConvS2S [8]	26:36	41,29	7,7 · 1019	1,2 · 1021
Máy biến áp (mô hình cơ sở)	27,3	38,1	3,3 · 1018	
Máy biến áp (lớn)	28,4	41,0	2.3 · 1019	

Làm mịn nhãn Trong quá trình đào tạo, chúng tôi đã sử dụng làm mịn nhãn có giá trị ls = 0,1 [30]. Cái này
gây ra sự bối rối vì mô hình trở nên không chắc chắn hơn như ng cải thiện độ chính xác và điểm BLEU.

6 kết quả

6.1 Dịch máy

Về nhiệm vụ dịch thuật từ Anh sang Đức của WMT 2014, mô hình máy biến áp lớn (Transformer (big)
trong Bảng 2) vư ợt trội hơn các mô hình đư ợc báo cáo tốt nhất trư ớc đây (bao gồm cả các nhóm) hơn 2,0
BLEU, thiết lập điểm BLEU hiện đại mới là 28,4. Cấu hình của model này là
đư ợc liệt kê ở dòng dư ới cùng của Bảng 3. Quá trình đào tạo mất 3,5 ngày trên 8 GPU P100. Ngay cả mô hình cơ sở của chúng tôi
vư ợt qua tất cả các mô hình và tổ hợp đã đư ợc công bố trư ớc đó, với chi phí đào tạo chỉ bằng một phần nhỏ của bất kỳ mô hình và tổ hợp nào
các mô hình cạnh tranh

Trong nhiệm vụ dịch thuật từ tiếng Anh sang tiếng Pháp của WMT 2014, mô hình lớn của chúng tôi đạt đư ợc số điểm BLEU là 41,0,
vư ợt trội hơn tất cả các mô hình đơn lẻ đã đư ợc công bố trư ớc đó, với chi phí đào tạo thấp hơn 1/4
mô hình tiên tiến trư ớc đó. Model Transformer (lớn) đư ợc huấn luyện dịch từ Anh sang Pháp đư ợc sử dụng
tỷ lệ bỏ học Pdrop = 0,1, thay vì 0,3.

Đối với các mô hình cơ sở, chúng tôi đã sử dụng một mô hình duy nhất thu đư ợc bằng cách lấy trung bình 5 điểm kiểm tra cuối cùng.
đư ợc viết cách nhau 10 phút. Đối với các mô hình lớn, chúng tôi tính trung bình 20 điểm kiểm tra cuối cùng. Chúng tôi
đã sử dụng tìm kiếm chùm tia với kích thư ớc chùm tia là 4 và độ dài bị phạt α = 0,6 [31]. Các siêu tham số này
đư ợc chọn sau khi thử nghiệm trên tập phát triển. Chúng tôi đặt độ dài đầu ra tối đa trong
suy luận độ dài đầu vào +50, như ng kết thúc sớm khi có thể [31].

Bảng 2 tóm tắt kết quả của chúng tôi và so sánh chất lư ợng dịch thuật và chi phí đào tạo của chúng tôi với mô hình khác
kiến trúc từ văn học. Chúng tôi ư ớc tính số lư ợng phép toán dấu phẩy động đư ợc sử dụng để huấn luyện một
mô hình bằng cách nhân thời gian đào tạo, số lư ợng GPU đư ợc sử dụng và ư ớc tính khả năng duy trì
dung lư ợng dấu phẩy động có độ chính xác đơn của mỗi GPU [5].

6.2 Các biến thể của mô hình

Để đánh giá tầm quan trọng của các thành phần khác nhau của Máy biến áp, chúng tôi đã thay đổi mô hình cơ sở của mình
theo những cách khác nhau, đo lư ờng sự thay đổi về hiệu suất của bản dịch từ tiếng Anh sang tiếng Đức trên
bộ phát triển, newstest2013. Chúng tôi đã sử dụng tìm kiếm chùm tia như đư ợc mô tả trong phần trư ớc, như ng không
điểm kiểm tra trung bình. Chúng tôi trình bày những kết quả này trong Bảng 3.

Trong Bảng 3 hàng (A), chúng tôi thay đổi số lư ợng đầu chú ý cũng như kích thư ớc khóa và giá trị chú ý,
giữ lư ợng tính toán không đổi, như đư ợc mô tả trong Phần 3.2.2. Trong khi đầu đơn
sự chú ý kém hơn 0,9 BLEU so với cài đặt tốt nhất, chất lư ợng cũng giảm sút khi có quá nhiều đầu.

5Chúng tôi sử dụng các giá trị 2,8, 3,7, 6,0 và 9,5 TFLOPS tư ơng ứng cho K80, K40, M40 và P100.

Bảng 3: Các biến thể của kiến trúc máy biến áp. Các giá trị không được liệt kê giống hệt với các giá trị cơ sở ngư ời mẫu. Tất cả số liệu đều có trong bộ phát triển dịch thuật từ tiếng Anh sang tiếng Đức, newstest2013. Liệt kê sự phức tạp xảy ra trên mỗi từ, theo mã hóa cặp byte của chúng tôi và không nên so sánh với sự phức tạp của mỗi từ.

	mô hình N	dff	h	dk	dv	Pdrop	ls	đào tạo bư ớc	thông số (dev)	PPL (dev)	BLEU 25,8	×106
cơ sở	6	512	2048	8	64	64	0,1	0,1 100K	4,92	5,29	24,9	65
(MỘT)			1	512	512	4	128			5,00		
			128	16	32	32	32			4,91	25,5	
			16	16						5,01	25,8	
											25,4	
(B)						16				5,16	25,1	58
						32				5,01	25,4	60
(C)	2									6,11	23,7	36
	4									5,19	25,3	50
	8									4,88	25,5	80
		256			32	32	128			5,75	24,5	28
		1024			128					4,66	26,0	168
			1024							5,12	25,4	53
			4096							4,75	26,2	90
(D)							0,0			5,77	24,6	
							0,2			4,95	25,5	
								0,0		4,67	25,3	
								0,2		5,47	25,7	
(E)	nhúng vị trí thay vì hình sin									4,92	25,7	
lớn	6	1024	4096	16			0,3	300K	4,33		26,4	213

Trong Bảng 3 hàng (B), chúng tôi nhận thấy rằng việc giảm kích thư ớc khóa chú ý dk sẽ ảnh hư ởng đến chất lư ợng mô hình. Cái này gợi ý rằng việc xác định khả năng tư ơng thích không dễ dàng và khả năng tư ơng thích phức tạp hơn chức năng hơn sản phẩm chấm có thể có lợi. Chúng tôi quan sát thêm ở hàng (C) và (D) rằng, như mong đợi, các mô hình lớn hơn thì tốt hơn và việc bỏ qua rất hữu ích trong việc tránh lấp quá mức. Trong hàng (E), chúng tôi thay thế mã hóa vị trí hình sin với các phần nhúng vị trí đã học [8] và quan sát gần như giống hệt nhau kết quả cho mô hình cơ sở.

7. Kết luận

Trong công việc này, chúng tôi đã trình bày Transformer, mô hình truyền tải trình tự đầu tiên hoàn toàn dựa trên chú ý, thay thế các lớp lặp lại đư ợc sử dụng phổ biến nhất trong kiến trúc bộ mã hóa-giải mã bằng tự chú ý nhiều đầu.

Đối với các tác vụ dịch thuật, Transformer có thể đư ợc huấn luyện nhanh hơn đáng kể so với các tác vụ dựa trên kiến trúc. trên các lớp hồi quy hoặc tích chập. Trên cả WMT 2014 tiếng Anh sang tiếng Đức và WMT 2014 Nhiệm vụ dịch thuật từ tiếng Anh sang tiếng Pháp, chúng tôi đạt đư ợc một trạng thái nghệ thuật mới. Trong nhiệm vụ trư ớc đây, điều tốt nhất của chúng tôi mô hình thậm chí còn vư ợt trội hơn tất cả các nhóm đư ợc báo cáo trư ớc đó.

Chúng tôi rất vui mừng về tư ơng lai của các mô hình dựa trên sự chú ý và có kế hoạch áp dụng chúng vào các nhiệm vụ khác. Chúng tôi kế hoạch mở rộng Transformer cho các vấn đề liên quan đến phư ơng thức đầu vào và đầu ra ngoài văn bản và để điều tra các cơ chế chú ý hạn chế, cục bộ để xử lý hiệu quả đầu vào và đầu ra lớn như hình ảnh, âm thanh và video. Làm cho thế hệ ít tuần tự hơn là một mục tiêu nghiên cứu khác của chúng tôi.

Mã chúng tôi sử dụng để đào tạo và đánh giá các mô hình của mình có sẵn tại https://github.com/tensorflow/tensor2tensor.

Lời cảm ơn Chúng tôi rất biết ơn Nal Kalchbrenner và Stephan Gouws vì sự thành công của họ nhận xét, sửa chữa và cảm hứng.

[1] Jimmy Lei Ba, Jamie Ryan Kiros và Geoffrey E Hinton. Chuẩn hóa lớp. bản in trư ớc arXiv arXiv :1607.06450, 2016.

[2] Dzmitry Bahdanau, Kyunghyun Cho và Yoshua Bengio. Dịch máy thần kinh bằng cách cùng nhau học cách căn chỉnh và dịch. CoRR, abs/1409.0473, 2014.

[3] Denny Britz, Anna Goldie, Minh-Thang Lư ơng, và Quốc V. Lê. Khám phá lớn về thần kinh kiến trúc dịch máy CoRR, abs/1703.03906, 2017.

[4] Jianpeng Cheng, Li Dong và Mirella Lapata. Mạng bộ nhớ ngắn hạn dài cho máy đọc. bản in trư ớc arXiv arXiv:1601.06733, 2016.

[5] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk và Yoshua Bengio. Học cách biểu diễn cụm từ bằng cách sử dụng bộ mã hóa-giải mã rnn để dịch máy thống kê. CoRR, abs/1406.1078, 2014.

[6] Francois Chollet. Xception: Học sâu với các tích chập có thể phân tách theo chiều sâu. arXiv bản in trư ớc arXiv:1610.02357, 2016.

[7] Junyoung Chung, Ç aglar Gülçehre, Kyunghyun Cho và Yoshua Bengio. Đánh giá thực nghiệm của các mạng thần kinh tái phát có kiểm soát trên mô hình trình tự. CoRR, abs/1412.3555, 2014.

[8] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats và Yann N. Dauphin. Trình tự liên hợp để học theo trình tự. bản in trư ớc arXiv arXiv:1705.03122v2, 2017.

[9] Alex Graves. Tạo chuỗi với mạng lư ới thần kinh tái phát. bản in trư ớc arXiv arXiv:1308.0850, 2013.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren và Jian Sun. Học phần dư sâu để nhận dạng độ tuổi hình ảnh. Trong Kỷ yếu của Hội nghị IEEE về Thị giác máy tính và Nhận dạng mẫu, trang 770–778, 2016.

[11] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi và Jürgen Schmidhuber. Dòng chuyển màu trong Mạng hồi quy: khó khăn trong việc học các mối phụ thuộc dài hạn, 2001.

[12] Sepp Hochreiter và Jürgen Schmidhuber. Trí nhớ ngắn hạn dài. Tính toán thần kinh, 9(8):1735–1780, 1997.

[13] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer và Yonghui Wu. Khám phá các giới hạn của mô hình ngôn ngữ. bản in trư ớc arXiv arXiv:1602.02410, 2016.

[14] Łukasz Kaiser và Ilya Sutskever. GPU thần kinh học các thuật toán. Trong Hội nghị quốc tế về đại diện học tập (ICLR), 2016.

[15] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves và Ko-ray Kavukcuoglu. Dịch máy thần kinh trong thời gian tuyến tính. bản in trư ớc arXiv arXiv:1610.10099v2, 2017.

[16] Yoon Kim, Carl Denton, Lư ơng Hoàng và Alexander M. Rush. Mạng lư ới chú ý có cấu trúc. Trong Hội nghị quốc tế về đại diện học tập, 2017.

[17] Diederik Kingma và Jimmy Ba. Adam: Một phư ơng pháp tối ư u hóa ngẫu nhiên. Trong ICLR, 2015.

[18] Oleksii Kuchaiev và Boris Ginsburg. Thủ thuật nhân tố hóa cho mạng LSTM. bản in trư ớc arXiv arXiv :1703.10722, 2017.

[19] Chu Hàn Lâm, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Chu, và Yoshua Bengio. Một câu nhúng có cấu trúc chú ý đến bản thân. bản in trư ớc arXiv arXiv :1703.03130, 2017.

[20] Samy Bengio Łukasz Kaiser. Bộ nhớ hoạt động có thể thay thế sự chú ý? Những tiến bộ trong thần kinh Hệ thống xử lý thông tin, (NIPS), 2016.

[21] Minh-Thang Lương, Hiếu Phạm, và Christopher D Manning. Các phương pháp tiếp cận hiệu quả đối với dịch máy thần kinh dựa trên sự chú ý. bản in trước arXiv arXiv:1508.04025, 2015.

[22] Ankur Parikh, Oscar Täckström, Dipanjan Das, và Jakob Uszkoreit. Một mô hình chú ý có thể phân hủy. Trong Phương pháp thực nghiệm trong xử lý ngôn ngữ tự nhiên, 2016.

[23] Romain Paulus, Caiming Xiong, và Richard Socher. Một mô hình được củng cố sâu sắc cho tính trừu tượng tóm tắt. bản in trước arXiv arXiv:1705.04304, 2017.

[24] Ofir Press và Lior Wolf. Sử dụng tính năng nhúng đầu ra để cải thiện các mô hình ngôn ngữ. arXiv bản in trước arXiv:1608.05859, 2016.

[25] Rico Sennrich, Barry Haddow và Alexandra Birch. Dịch máy thần kinh các từ hiếm với các đơn vị từ phụ. bản in trước arXiv arXiv:1508.07909, 2015.

[26] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quốc Lê, Geoffrey Hinton và Jeff Dean. Mạng lưới thần kinh cực kỳ lớn: Lớp hỗn hợp các chuyên gia có cổng thưa thớt . bản in trước arXiv arXiv:1701.06538, 2017.

[27] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, và Ruslan Salakhutdi- nov. Bỏ học: một cách đơn giản để ngăn chặn mạng lưới thần kinh bị trang bị quá mức. Tạp chí Nghiên cứu Học máy, 15(1):1929–1958, 2014.

[28] Sainbayar Sukhbaatar, arthur szlam, Jason Weston và Rob Fergus. Mạng bộ nhớ đầu cuối. Trong C. Cortes, ND Lawrence, DD Lee, M. Sugiyama và R. Garnett, các biên tập viên, Những tiến bộ trong Hệ thống xử lý thông tin thần kinh 28, trang 2440–2448. Hiệp hội Curran, Inc., 2015.

[29] Ilya Sutskever, Oriol Vinyals, và Quốc VV Lê. Trình tự học theo trình tự với mạng lưới thần kinh. Trong Những tiến bộ trong hệ thống xử lý thông tin thần kinh, trang 3104–3112, 2014.

[30] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, và Zbigniew Wojna. Xem xét lại kiến trúc khởi đầu cho thị giác máy tính. CoRR, abs/1512.00567, 2015.

[31] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quốc V Lê, Mohammad Norouzi, Wolfgang Macherey , Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, và những người khác. Hệ thống dịch máy thần kinh của Google : Thu hẹp khoảng cách giữa bản dịch của con người và máy. bản in trước arXiv arXiv :1609.08144, 2016.

[32] Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, và Wei Xu. Các mô hình hồi quy sâu với các kết nối chuyển tiếp nhanh để dịch máy thần kinh. CoRR, abs/1606.04199, 2016.