Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

*Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

†Work performed while at Google Brain.

‡Work performed while at Google Research.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

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 tư ợng

Các mô hình truyền tải trình tự chiếm ư u 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 lư ợng vư ợt 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 đư ợc 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 tiên tiến nhất cho một mô hình mới là 41,8 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 Chúng tôi cho thấy rằng Transformer khái quát hóa tốt các nhiệm vụ khác bằng cách áp dụng thành công nó vào phân tích cú pháp khu vực bầu cử tiếng Anh với cả dữ liệu đào tạo lớn và hạn chế.

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 tư ởng 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 ngư ời 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 trư ớc đó của chúng tôi, cải thiện đáng kể kết quả và tăng tốc đáng kể nghiên cứu của chúng tôi.

†Công việc đư ợc thực hiện khi làm việc tại Google Brain. ‡Công việc đư ợc 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.

1 Introduction

Recurrent neural networks, long short-term memory [13] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [35, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [38, 24, 15].

Recurrent models typically factor computation along the symbol positions of the input and output sequences. Aligning the positions to steps in computation time, they generate a sequence of hidden states h_t, as a function of the previous hidden state h_{t-1} and the input for position t. This inherently sequential nature precludes parallelization within training examples, which becomes critical at longer sequence lengths, as memory constraints limit batching across examples. Recent work has achieved significant improvements in computational efficiency through factorization tricks [21] and conditional computation [32], while also improving model performance in case of the latter. The fundamental constraint of sequential computation, however, remains.

Attention mechanisms have become an integral part of compelling sequence modeling and transduction models in various tasks, allowing modeling of dependencies without regard to their distance in the input or output sequences [2, 19]. In all but a few cases [27], however, such attention mechanisms are used in conjunction with a recurrent network.

In this work we propose the Transformer, a model architecture eschewing recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output. The Transformer allows for significantly more parallelization and can reach a new state of the art in translation quality after being trained for as little as twelve hours on eight P100 GPUs.

2 Background

The goal of reducing sequential computation also forms the foundation of the Extended Neural GPU [16], ByteNet [18] and ConvS2S [9], all of which use convolutional neural networks as basic building block, computing hidden representations in parallel for all input and output positions. In these models, the number of operations required to relate signals from two arbitrary input or output positions grows in the distance between positions, linearly for ConvS2S and logarithmically for ByteNet. This makes it more difficult to learn dependencies between distant positions [12]. In the Transformer this is reduced to a constant number of operations, albeit at the cost of reduced effective resolution due to averaging attention-weighted positions, an effect we counteract with Multi-Head Attention as described in section 3.2.

Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. Self-attention has been used successfully in a variety of tasks including reading comprehension, abstractive summarization, textual entailment and learning task-independent sentence representations [4, 27, 28, 22].

End-to-end memory networks are based on a recurrent attention mechanism instead of sequence-aligned recurrence and have been shown to perform well on simple-language question answering and language modeling tasks [34].

To the best of our knowledge, however, the Transformer is the first transduction model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution. In the following sections, we will describe the Transformer, motivate self-attention and discuss its advantages over models such as [17, 18] and [9].

3 Model Architecture

Most competitive neural sequence transduction models have an encoder-decoder structure [5, 2, 35]. Here, the encoder maps an input sequence of symbol representations $(x_1, ..., x_n)$ to a sequence of continuous representations $\mathbf{z} = (z_1, ..., z_n)$. Given $\mathbf{z}$, the decoder then generates an output sequence $(y_1, ..., y_m)$ of symbols one element at a time. At each step the model is auto-regressive [10], consuming the previously generated symbols as additional input when generating the next.

1. Giới thiệu

Mạng thần kinh tái phát, bộ nhớ ngắn hạn dài [13] 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 về mô hình hóa và chuyển đổi trình tự như mô hình ngôn ngữ và dịch máy [35, 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 [38, 24, 15].

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 [21] và tính toán có điều kiện [32], 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 hóa 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 chuỗi đầu vào hoặc đầu ra [2, 19]. Tuy nhiên, trong tất cả trừ một số trường hợp [27], 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 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 [16], ByteNet [18] và ConvS2S [9], 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 [12]. 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. 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, 27, 28, 22].

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 [34].

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 [17, 18] và [9].

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, 35]. 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 [10], sử dụng các ký hiệu được tạo trước đó làm đầu vào bổ sung khi tạo ký hiệu tiếp theo.

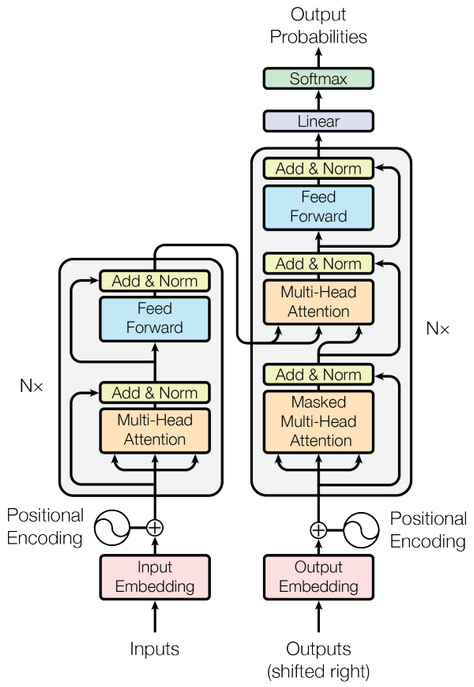


Figure 1: The Transformer - model architecture.

The Transformer follows this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, shown in the left and right halves of Figure 1, respectively.

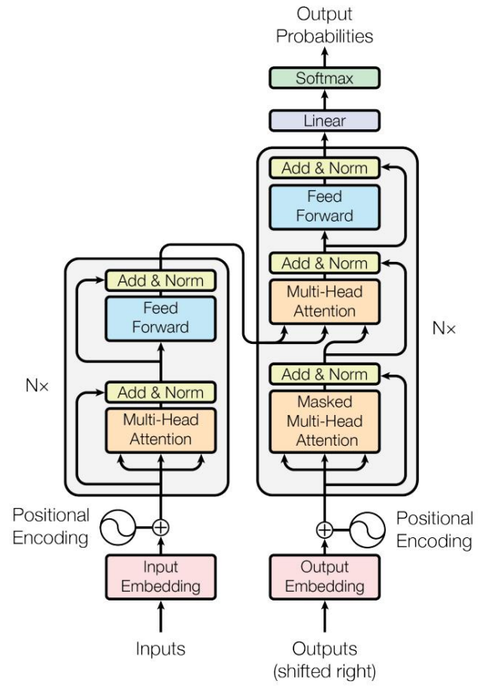
3.1 Encoder and Decoder Stacks

Encoder: The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection [11] around each of the two sub-layers, followed by layer normalization [1]. That is, the output of each sub-layer is $\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$, where $\mathrm{Sublayer}(x)$ is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension $d_{\text{model}} = 512$.

Decoder: The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with fact that the output embeddings are offset by one position, ensures that the predictions for position i can depend only on the known outputs at positions less than i.

3.2 Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum



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

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, đư ợc kết nối đầy đủ cho cả bộ mã hóa và bộ giải mã, tư ơng ứng đư ợc 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 mạng chuyển tiếp nguồn cấp dữ liệu đư ợc kết nối đầy đủ, đơn giản theo vị trí. Chúng tôi sử dụng kết nối còn lại [11] 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 thư ớc 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. Tư ơng 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 đư ợc 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 đư ợc 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 đư ợc tính dư ới dạng tổng có trọng số

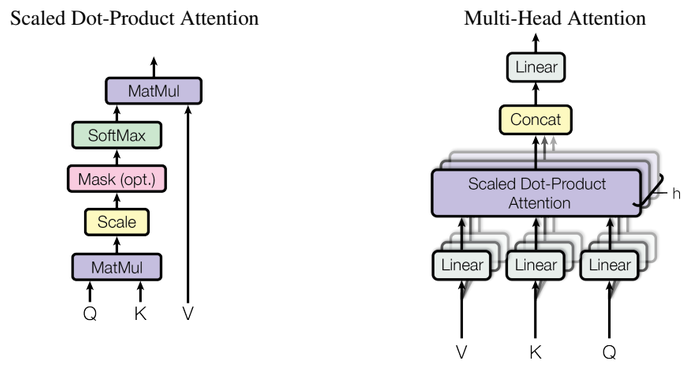


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

3.2.1 Scaled Dot-Product Attention

We call our particular attention "Scaled Dot-Product Attention" (Figure 2). The input consists of queries and keys of dimension d_k, and values of dimension d_v. We compute the dot products of the query with all keys, divide each by $\sqrt{d_k}$, and apply a softmax function to obtain the weights on the values.

In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V. We compute the matrix of outputs as:

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

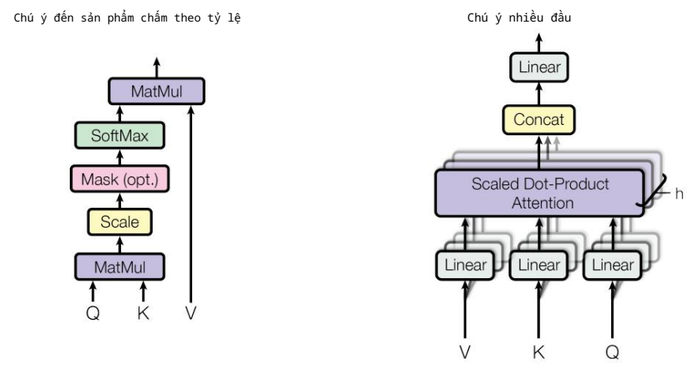
The two most commonly used attention functions are additive attention [2], and dot-product (multiplicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of $\frac{1}{\sqrt{d_k}}$. Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code.

While for small values of d_k the two mechanisms perform similarly, additive attention outperforms dot product attention without scaling for larger values of d_k [3]. We suspect that for large values of d_k, the dot products grow large in magnitude, pushing the softmax function into regions where it has extremely small gradients [4]. To counteract this effect, we scale the dot products by $\frac{1}{\sqrt{d_k}}$.

3.2.2 Multi-Head Attention

Instead of performing a single attention function with d_{model}-dimensional keys, values and queries, we found it beneficial to linearly project the queries, keys and values h times with different, learned linear projections to d_k, d_k and d_v dimensions, respectively. On each of these projected versions of queries, keys and values we then perform the attention function in parallel, yielding d_v-dimensional

[4]To illustrate why the dot products get large, assume that the components of q and k are independent random variables with mean 0 and variance 1. Then their dot product, $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$, has mean 0 and variance d_k.



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.

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 tích số chấm của truy vấn bằng tất cả các khóa, chia từng khóa cho √ dk 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:

$$\text{Chú ý(Q, K, V) = softmax(QKT} \quad \frac{}{\sqrt{\text{dk}}})\text{V} \quad (1)$$

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 $\frac{1}{\sqrt{dk}}$. 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, nhưng 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 dk 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 dk [3]. Chúng tôi nghi ngờ rằng đối với các giá trị lớn của dk, 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 $\frac{1}{\sqrt{dk}}$.

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 sẽ có lợi khi 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. Sau đó, trên mỗi phiên bản truy vấn, khóa và giá trị dự kiến này, chúng tôi thực hiện song song chức năng chú ý, mang lại dv chiều

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

output values. These are concatenated and once again projected, resulting in the final values, as depicted in Figure 2.

Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this.

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_\text{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_\text{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_\text{model} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_\text{model}}$.

In this work we employ $h = 8$ parallel attention layers, or heads. For each of these we use $d_k = d_v = d_\text{model}/h = 64$. Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality.

3.2.3 Applications of Attention in our Model

The Transformer uses multi-head attention in three different ways:

- In "encoder-decoder attention" layers, the queries come from the previous decoder layer, and the memory keys and values come from the output of the encoder. This allows every position in the decoder to attend over all positions in the input sequence. This mimics the typical encoder-decoder attention mechanisms in sequence-to-sequence models such as [38, 2, 9].
- The encoder contains self-attention layers. In a self-attention layer all of the keys, values and queries come from the same place, in this case, the output of the previous layer in the encoder. Each position in the encoder can attend to all positions in the previous layer of the encoder.
- Similarly, self-attention layers in the decoder allow each position in the decoder to attend to all positions in the decoder up to and including that position. We need to prevent leftward information flow in the decoder to preserve the auto-regressive property. We implement this inside of scaled dot-product attention by masking out (setting to $-\infty$) all values in the input of the softmax which correspond to illegal connections. See Figure 2.

3.3 Position-wise Feed-Forward Networks

In addition to attention sub-layers, each of the layers in our encoder and decoder contains a fully connected feed-forward network, which is applied to each position separately and identically. This consists of two linear transformations with a ReLU activation in between.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (2)$$

While the linear transformations are the same across different positions, they use different parameters from layer to layer. Another way of describing this is as two convolutions with kernel size 1. The dimensionality of input and output is $d_\text{model} = 512$, and the inner-layer has dimensionality $d_{ff} = 2048$.

3.4 Embeddings and Softmax

Similarly to other sequence transduction models, we use learned embeddings to convert the input tokens and output tokens to vectors of dimension d_model. We also use the usual learned linear transformation and softmax function to convert the decoder output to predicted next-token probabilities. In our model, we share the same weight matrix between the two embedding layers and the pre-softmax linear transformation, similar to [30]. In the embedding layers, we multiply those weights by $\sqrt{d_\text{model}}$.

các giá trị đầu ra. 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.

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

headi = Chú ý(QWQ tôi, KW K tôi, V WV)

Trong đó các phép chiếu là ma trận tham số W và Q R dmodel×dk , W K R dmodel×dk , WV R dmodel×dv 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 [38, 2, 9].
- 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 (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 [30]. Trong các lớp nhúng, chúng tôi nhân các trọng số đó với √ dmodel.

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

3.5 Positional Encoding

Since our model contains no recurrence and no convolution, in order for the model to make use of the order of the sequence, we must inject some information about the relative or absolute position of the tokens in the sequence. To this end, we add "positional encodings" to the input embeddings at the bottoms of the encoder and decoder stacks. The positional encodings have the same dimension d_{model} as the embeddings, so that the two can be summed. There are many choices of positional encodings, learned and fixed [9].

In this work, we use sine and cosine functions of different frequencies:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$. We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}.

We also experimented with using learned positional embeddings [9] instead, and found that the two versions produced nearly identical results (see Table 3 row (E)). We chose the sinusoidal version because it may allow the model to extrapolate to sequence lengths longer than the ones encountered during training.

4 Why Self-Attention

In this section we compare various aspects of self-attention layers to the recurrent and convolutional layers commonly used for mapping one variable-length sequence of symbol representations $(x_1, ..., x_n)$ to another sequence of equal length $(z_1, ..., z_n)$, with $x_i, z_i \in \mathbb{R}^d$, such as a hidden layer in a typical sequence transduction encoder or decoder. Motivating our use of self-attention we consider three desiderata.

One is the total computational complexity per layer. Another is the amount of computation that can be parallelized, as measured by the minimum number of sequential operations required.

The third is the path length between long-range dependencies in the network. Learning long-range dependencies is a key challenge in many sequence transduction tasks. One key factor affecting the ability to learn such dependencies is the length of the paths forward and backward signals have to traverse in the network. The shorter these paths between any combination of positions in the input and output sequences, the easier it is to learn long-range dependencies [12]. Hence we also compare the maximum path length between any two input and output positions in networks composed of the different layer types.

As noted in Table 1, a self-attention layer connects all positions with a constant number of sequentially executed operations, whereas a recurrent layer requires $O(n)$ sequential operations. In terms of computational complexity, self-attention layers are faster than recurrent layers when the sequence

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

Loại lớp	Độ phức tạp trên mỗi lớp	Độ dài đư ờng dẫn tối đa tuần tự Hoạt động	
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)

3.5 Mã hóa vị trí

Vì mô hình của chúng tôi không có phép lặp và không có tích chập, để mô hình có thể sử dụng thứ tự của dãy, chúng ta phải đư a vào một số thông tin về vị trí tư ơng đối hoặc tuyệt đối của token trong chuỗi. Để đạt đư ợc mục đích này, chúng tôi thêm "mã hóa vị trí" vào phần nhúng đầu vào tại 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 đư ợc tóm tắt. Có nhiều lựa chọn về mã hóa vị trí, đã học và sửa chữa [9].

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 thư ớc. Nghĩa là, mỗi chiều của mã hóa vị trí tư ơng ứng với một hình sin. Các bư ớc sóng tạo thành một cấp số nhân từ 2π đến 10000 · 2π. Chúng tôi đã chọn chức năng này vì chúng tôi đư a ra giả thuyết rằng nó sẽ cho phép mô hình dễ dàng học cách tham dự các vị trí tư ơng đối, vì đối với bất kỳ độ lệch k cố định nào, P Epos+k có thể đư ợc biểu diễn dư ới 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 [9] 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 thư ờng đư ợc 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 lư ợng tính toán có thể đư ợc song song hóa, đư ợc đo bằng số lư ợng hoạt động tuần tự tối thiểu đư ợc yêu cầu.

Thứ ba là độ dài đư ờng 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 hư ởng đến khả năng tìm hiểu các phụ thuộc như vậy là độ dài của các đư ờng dẫn tín hiệu tiến và lùi phải có truyền trong mạng. Các đư ờng 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 [12]. Do đó chúng tôi cũng so sánh độ dài đư ờng 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 lư u ý trong Bảng 1, lớp tự chú ý kết nối tất cả các vị trí với số lư ợng liên tục không đổi các hoạt động đư ợc 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

length n is smaller than the representation dimensionality d, which is most often the case with sentence representations used by state-of-the-art models in machine translations, such as word-piece [38] and byte-pair [31] representations. To improve computational performance for tasks involving very long sequences, self-attention could be restricted to considering only a neighborhood of size r in the input sequence centered around the respective output position. This would increase the maximum path length to $O(n/r)$. We plan to investigate this approach further in future work.

A single convolutional layer with kernel width $k < n$ does not connect all pairs of input and output positions. Doing so requires a stack of $O(n/k)$ convolutional layers in the case of contiguous kernels, or $O(log_k(n))$ in the case of dilated convolutions [18], increasing the length of the longest paths between any two positions in the network. Convolutional layers are generally more expensive than recurrent layers, by a factor of k. Separable convolutions [6], however, decrease the complexity considerably, to $O(k \cdot n \cdot d + n \cdot d^2)$. Even with $k = n$, however, the complexity of a separable convolution is equal to the combination of a self-attention layer and a point-wise feed-forward layer, the approach we take in our model.

As side benefit, self-attention could yield more interpretable models. We inspect attention distributions from our models and present and discuss examples in the appendix. Not only do individual attention heads clearly learn to perform different tasks, many appear to exhibit behavior related to the syntactic and semantic structure of the sentences.

5 Training

This section describes the training regime for our models.

5.1 Training Data and Batching

We trained on the standard WMT 2014 English-German dataset consisting of about 4.5 million sentence pairs. Sentences were encoded using byte-pair encoding [3], which has a shared source-target vocabulary of about 37000 tokens. For English-French, we used the significantly larger WMT 2014 English-French dataset consisting of 36M sentences and split tokens into a 32000 word-piece vocabulary [38]. Sentence pairs were batched together by approximate sequence length. Each training batch contained a set of sentence pairs containing approximately 25000 source tokens and 25000 target tokens.

5.2 Hardware and Schedule

We trained our models on one machine with 8 NVIDIA P100 GPUs. For our base models using the hyperparameters described throughout the paper, each training step took about 0.4 seconds. We trained the base models for a total of 100,000 steps or 12 hours. For our big models,(described on the bottom line of table 3), step time was 1.0 seconds. The big models were trained for 300,000 steps (3.5 days).

5.3 Optimizer

We used the Adam optimizer [20] with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$. We varied the learning rate over the course of training, according to the formula:

$$lrate = d_{\text{model}}^{-0.5} \cdot \min(step\_num^{-0.5}, step\_num \cdot warmup\_steps^{-1.5}) \quad (3)$$

This corresponds to increasing the learning rate linearly for the first $warmup\_steps$ training steps, and decreasing it thereafter proportionally to the inverse square root of the step number. We used $warmup\_steps = 4000$.

5.4 Regularization

We employ three types of regularization during training:

độ dài n nhỏ hơn chiều biểu diễn d, điều này thư ờng xảy ra với các biểu diễn câu đư ợc sử dụng bởi các mô hình hiện đại trong các bản dịch máy, chẳng hạn như biểu diễn từng từ [38] và cặp byte [31] . Để cải thiện hiệu suất tính toán cho các tác vụ liên quan đến chuỗi rất dài, việc tự chú ý có thể bị hạn chế khi chỉ xem xét một vùng lân cận có kích thư ớc r trong chuỗi đầu vào tập trung quanh vị trí đầu ra tư ơng ứng. Điều này sẽ tăng độ dài đư ờng dẫn tối đa lên O(n/r). Chúng tôi dự định điều tra phư ơng pháp này hơn nữa trong công việc trong tư ơng 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 trư ờng hợp các hạt nhân liền kề hoặc O(logk(n)) trong trư ờng hợp các lớp chập bị giãn [18], tăng độ dài của các đư ờng đi dài nhất giữa hai vị trí bất kỳ trong mạng. Các lớp tích chập thư ờng 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 tư ơng đư ơng 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 đư ợc 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 ngư ời chú ý riêng lẻ không chỉ học cách thực hiện các nhiệm vụ khác nhau một cách rõ ràng mà nhiều ngư ời 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 đư ợc 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 các mã thông báo thành một từ vựng gồm 32000 từ [38]. Các cặp câu đư ợc 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 đư ợc mô tả trong suốt bài viết, mỗi bư ớc huấn luyện mất khoảng 0,4 giây. Chúng tôi đã huấn luyện các mô hình cơ sở với tổng số 100.000 bư ớc hoặc 12 giờ. Đối với các mô hình lớn của chúng tôi (đư ợc mô tả ở dòng cuối cùng của bảng 3), thời gian bư ớc là 1,0 giây. Những ngư ời mẫu lớn đư ợc huấn luyện 300.000 bư ớc (3,5 ngày).

5.3 Trình tối ư u hóa

Chúng tôi đã sử dụng trình tối ư u hóa Adam [20] với β1 = 0,9, β2 = 0,98 và ε = 10 9 . Chúng tôi thay đổi tốc độ học tập trong suốt quá trình đào tạo, theo công thức:

$$\text{lrate} = \text{d}_{\text{ngư ời mẫu}}^{0,5} \cdot \text{phút(bư ớc\_num } 0,5 \text{ , bư ớc\_num} \cdot \text{ bư ớc khởi động } 1,5 \text{)} \quad (3)$$

Điều này tư ơng ứng với việc tăng tốc độ học tập một cách tuyến tính cho các bư ớc 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 bư ớc. 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:

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Residual Dropout We apply dropout [33] to the output of each sub-layer, before it is added to the sub-layer input and normalized. In addition, we apply dropout to the sums of the embeddings and the positional encodings in both the encoder and decoder stacks. For the base model, we use a rate of $P_{drop} = 0.1$.

Label Smoothing During training, we employed label smoothing of value $\epsilon_{ls} = 0.1$ [36]. This hurts perplexity, as the model learns to be more unsure, but improves accuracy and BLEU score.

6 Results

6.1 Machine Translation

On the WMT 2014 English-to-German translation task, the big transformer model (Transformer (big) in Table 2) outperforms the best previously reported models (including ensembles) by more than 2.0 BLEU, establishing a new state-of-the-art BLEU score of 28.4. The configuration of this model is listed in the bottom line of Table 3. Training took 3.5 days on 8 P100 GPUs. Even our base model surpasses all previously published models and ensembles, at a fraction of the training cost of any of the competitive models.

On the WMT 2014 English-to-French translation task, our big model achieves a BLEU score of 41.0, outperforming all of the previously published single models, at less than $1/4$ the training cost of the previous state-of-the-art model. The Transformer (big) model trained for English-to-French used dropout rate $P_{drop} = 0.1$, instead of 0.3.

For the base models, we used a single model obtained by averaging the last 5 checkpoints, which were written at 10-minute intervals. For the big models, we averaged the last 20 checkpoints. We used beam search with a beam size of 4 and length penalty $\alpha = 0.6$ [38]. These hyperparameters were chosen after experimentation on the development set. We set the maximum output length during inference to input length + 50, but terminate early when possible [38].

Table 2 summarizes our results and compares our translation quality and training costs to other model architectures from the literature. We estimate the number of floating point operations used to train a model by multiplying the training time, the number of GPUs used, and an estimate of the sustained single-precision floating-point capacity of each GPU [5].

6.2 Model Variations

To evaluate the importance of different components of the Transformer, we varied our base model in different ways, measuring the change in performance on English-to-German translation on the

[5]We used values of 2.8, 3.7, 6.0 and 9.5 TFLOPS for K80, K40, M40 and P100, respectively.

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 23,75	EN-DE	EN-FR
ByteNet [18]				
Deep-Att + PosUnk [39]		39,2		1,0 · 1020
GNMT + RL [38]	24,6	39,92	2,3 · 1019	1,4 · 1020
Chuyển đổiS2S [9]	25,16	40,46	9,6 · 1018	1,5 · 1020
MoE [32]	26,03	40,56	2,0 · 1019	1,2 · 1020
Bộ đồng phục Deep-Att + PosUnk [39]		40,4		8,0 · 1020
Bộ hòa tấu GNMT + RL [38]	26:30	41,16	1,8 · 1020	1,1 · 1021
Nhóm ConvS2S [9]	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	41,8	2.3 · 1019	

Residual Dropout Chúng tôi áp dụng dropout [33] cho đầu ra của mỗi lớp con, trư ớc khi nó đư ợc thêm vào
đầu vào lớp con và chuẩn hóa. Ngoài ra, chúng tôi áp dụng bỏ học cho 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.

Làm mịn nhãn Trong quá trình đào tạo, chúng tôi đã sử dụng làm mịn nhãn có giá trị εls = 0,1 [36]. 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 [38]. 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 [38].

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 duy nhất 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

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.

Table 3: Variations on the Transformer architecture. Unlisted values are identical to those of the base model. All metrics are on the English-to-German translation development set, newstest2013. Listed perplexities are per-wordpiece, according to our byte-pair encoding, and should not be compared to per-word perplexities.

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	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				5.75	24.5	28
		1024			128	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)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

development set, newstest2013. We used beam search as described in the previous section, but no checkpoint averaging. We present these results in Table 3.

In Table 3 rows (A), we vary the number of attention heads and the attention key and value dimensions, keeping the amount of computation constant, as described in Section 3.2.2. While single-head attention is 0.9 BLEU worse than the best setting, quality also drops off with too many heads.

In Table 3 rows (B), we observe that reducing the attention key size d_k hurts model quality. This suggests that determining compatibility is not easy and that a more sophisticated compatibility function than dot product may be beneficial. We further observe in rows (C) and (D) that, as expected, bigger models are better, and dropout is very helpful in avoiding over-fitting. In row (E) we replace our sinusoidal positional encoding with learned positional embeddings [9], and observe nearly identical results to the base model.

6.3 English Constituency Parsing

To evaluate if the Transformer can generalize to other tasks we performed experiments on English constituency parsing. This task presents specific challenges: the output is subject to strong structural constraints and is significantly longer than the input. Furthermore, RNN sequence-to-sequence models have not been able to attain state-of-the-art results in small-data regimes [37].

We trained a 4-layer transformer with $d_{model} = 1024$ on the Wall Street Journal (WSJ) portion of the Penn Treebank [25], about 40K training sentences. We also trained it in a semi-supervised setting, using the larger high-confidence and BerkleyParser corpora from with approximately 17M sentences [37]. We used a vocabulary of 16K tokens for the WSJ only setting and a vocabulary of 32K tokens for the semi-supervised setting.

We performed only a small number of experiments to select the dropout, both attention and residual (section 5.4), learning rates and beam size on the Section 22 development set, all other parameters remained unchanged from the English-to-German base translation model. During inference, we

Bảng 3: Các biến thể của kiến trúc máy biến áp. Các giá trị không đư ợc 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	εls	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 lớn 6 1024 4096 16										4,92	25,7	
						0,3		300K	4,33		26,4	213

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.

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 lắp quá mức. Trong hàng (E), chúng tôi thay thế mã hóa vị trí hình sin với các phần nhúng vị trí đã học [9] và quan sát gần như giống hệt nhau kết quả cho mô hình cơ sở.

6.3 Phân tích cử tri tiếng Anh

Để đánh giá xem Transformer có thể khái quát hóa cho các nhiệm vụ khác hay không, chúng tôi đã thực hiện thí nghiệm trên tiếng Anh phân tích khu vực bầu cử. Nhiệm vụ này đư a ra những thách thức cụ thể: đầu ra phải chịu sự tác động mạnh mẽ của cơ cấu ràng buộc và dài hơn đáng kể so với đầu vào. Hơn nữa, trình tự RNN theo trình tự các mô hình đã không thể đạt đư ợc kết quả tiên tiến trong chế độ dữ liệu nhỏ [37].

Chúng tôi đã đào tạo máy biến áp 4 lớp với dmodel = 1024 trên phần Wall Street Journal (WSJ) của Penn Treebank [25], khoảng 40K câu huấn luyện. Chúng tôi cũng huấn luyện nó trong môi trư ờng bán giám sát, sử dụng tập tin có độ tin cậy cao và BerkleyParser lớn hơn với khoảng 17 triệu câu [37]. Chúng tôi đã sử dụng từ vựng gồm 16K mã thông báo cho cài đặt chỉ WSJ và từ vựng về 32K mã thông báo cho cài đặt bán giám sát.

Chúng tôi chỉ thực hiện một số thử nghiệm nhỏ để chọn ra ngư ời bỏ học, cả ngư ời chú ý và ngư ời còn lại. (phần 5.4), tốc độ học và kích thư ớc chùm tia trên bộ phát triển Phần 22, tất cả các tham số khác vẫn không thay đổi so với mô hình dịch cơ bản từ tiếng Anh sang tiếng Đức. Trong quá trình suy luận, chúng ta

Table 4: The Transformer generalizes well to English constituency parsing (Results are on Section 23 of WSJ)

Parser	Training	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37]	WSJ only, discriminative	88.3
Petrov et al. (2006) [29]	WSJ only, discriminative	90.4
Zhu et al. (2013) [40]	WSJ only, discriminative	90.4
Dyer et al. (2016) [8]	WSJ only, discriminative	91.7
Transformer (4 layers)	WSJ only, discriminative	91.3
Zhu et al. (2013) [40]	semi-supervised	91.3
Huang & Harper (2009) [14]	semi-supervised	91.3
McClosky et al. (2006) [26]	semi-supervised	92.1
Vinyals & Kaiser el al. (2014) [37]	semi-supervised	92.1
Transformer (4 layers)	semi-supervised	92.7
Luong et al. (2015) [23]	multi-task	93.0
Dyer et al. (2016) [8]	generative	93.3

increased the maximum output length to input length + 300. We used a beam size of 21 and $\alpha = 0.3$ for both WSJ only and the semi-supervised setting.

Our results in Table 4 show that despite the lack of task-specific tuning our model performs surprisingly well, yielding better results than all previously reported models with the exception of the Recurrent Neural Network Grammar [8].

In contrast to RNN sequence-to-sequence models [37], the Transformer outperforms the Berkeley-Parser [29] even when training only on the WSJ training set of 40K sentences.

7 Conclusion

In this work, we presented the Transformer, the first sequence transduction model based entirely on attention, replacing the recurrent layers most commonly used in encoder-decoder architectures with multi-headed self-attention.

For translation tasks, the Transformer can be trained significantly faster than architectures based on recurrent or convolutional layers. On both WMT 2014 English-to-German and WMT 2014 English-to-French translation tasks, we achieve a new state of the art. In the former task our best model outperforms even all previously reported ensembles.

We are excited about the future of attention-based models and plan to apply them to other tasks. We plan to extend the Transformer to problems involving input and output modalities other than text and to investigate local, restricted attention mechanisms to efficiently handle large inputs and outputs such as images, audio and video. Making generation less sequential is another research goals of ours.

The code we used to train and evaluate our models is available at `https://github.com/tensorflow/tensor2tensor`.

Acknowledgements We are grateful to Nal Kalchbrenner and Stephan Gouws for their fruitful comments, corrections and inspiration.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[3] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V. Le. Massive exploration of neural machine translation architectures. *CoRR*, abs/1703.03906, 2017.

[4] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

Bảng 4: Transformer tổng quát hóa tốt việc phân tích cú pháp cử tri tiếng Anh (Kết quả ở Mục 23 của WSJ)

Trình phân tích cú pháp	Đào tạo	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37] Chỉ WSJ, chỉ WSJ phân biệt đối xử,		88,3
Petrov và cộng sự. (2006) [29]	chỉ WSJ phân biệt đối xử,	90,4
Zhu và cộng sự. (2013) [40]	chỉ WSJ phân biệt đối xử,	90,4
Dyer và cộng sự. (2016) [8]	chỉ WSJ phân biệt đối xử,	91,7
Máy biến áp (4 lớp)	phân biệt đối xử bán giám	91,3
Zhu và cộng sự. (2013) [40]	sát bán giám sát	91,3
Hoàng & Harper (2009) [14]	bán giám sát bán	91,3
McClosky và cộng sự. (2006) [26]	giám sát tạo đa	92,1
Vinyals & Kaiser el al. (2014) [37]	tác vụ	92,1
Máy biến áp (4 lớp)		92,7
Lư ơng và cộng sự. (2015) [23]		93,0
Dyer và cộng sự. (2016) [8]		93,3

đã tăng độ dài đầu ra tối đa lên độ dài đầu vào + 300. Chúng tôi đã sử dụng kích thư ớc chùm tia là 21 và α = 0,3 chỉ dành cho cả WSJ và cài đặt bán giám sát.

Kết quả của chúng tôi trong Bảng 4 cho thấy rằng mặc dù thiếu sự điều chỉnh theo nhiệm vụ cụ thể như ng mô hình của chúng tôi hoạt động tốt một cách đáng ngạc nhiên, mang lại kết quả tốt hơn tất cả các mô hình đư ợc báo cáo trư ớc đó ngoại trừ mô hình. Ngữ pháp mạng thần kinh tái phát [8].

Ngư ợc lại với các mô hình tuần tự theo trình tự RNN [37], Transformer vư ợt trội hơn Berkeley- Parser [29] ngay cả khi chỉ đào tạo trên tập huấn luyện WSJ gồm 40K câu.

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/tenorflow/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.

Ngư ời giới thiệu

[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 sự phối hợp 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, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[6] Francois Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint arXiv:1610.02357*, 2016.

[7] Junyoung Chung, Çaglar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[8] Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. Recurrent neural network grammars. In *Proc. of NAACL*, 2016.

[9] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122v2*, 2017.

[10] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[12] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[14] Zhongqiang Huang and Mary Harper. Self-training PCFG grammars with latent annotations across languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 832–841. ACL, August 2009.

[15] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.

[16] Łukasz Kaiser and Samy Bengio. Can active memory replace attention? In *Advances in Neural Information Processing Systems, (NIPS)*, 2016.

[17] Łukasz Kaiser and Ilya Sutskever. Neural GPUs learn algorithms. In *International Conference on Learning Representations (ICLR)*, 2016.

[18] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099v2*, 2017.

[19] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. Structured attention networks. In *International Conference on Learning Representations*, 2017.

[20] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[21] Oleksii Kuchaiev and Boris Ginsburg. Factorization tricks for LSTM networks. *arXiv preprint arXiv:1703.10722*, 2017.

[22] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

[23] Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*, 2015.

[24] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[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] Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros và Noah A. Smith. Ngữ pháp mạng thần kinh tái phát . Trong Proc. của NAACL, 2016.

[9] 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.

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

[11] 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.

[12] 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.

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

[14] Zhongqiang Huang và Mary Harper. Ngữ pháp PCFG tự đào tạo với các chú thích tiềm ẩn trên các ngôn ngữ. Trong Kỷ yếu của Hội nghị năm 2009 về các phư ơng pháp thực nghiệm trong xử lý ngôn ngữ tự nhiên, trang 832–841. ACL, tháng 8 năm 2009.

[15] 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.

[16] Łukasz Kaiser và Samy Bengio. 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.

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

[18] 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.

[19] 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.

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

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

[22] 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.

[23] Minh-Thang Lư ơng, Quốc V. Lê, Ilya Sutskever, Oriol Vinyals, và Lukasz Kaiser. Trình tự đa nhiệm vụ để học theo trình tự. bản in trư ớc arXiv arXiv:1511.06114, 2015.

[24] Minh-Thang Lư ơng, Hiếu Phạm, và Christopher D Manning. Các phư ơng pháp tiếp cận hiệu quả đối với dịch máy thần kinh dựa trên sự chú ý. bản in trư ớc arXiv arXiv:1508.04025, 2015.

[25] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

[26] David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159. ACL, June 2006.

[27] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model. In *Empirical Methods in Natural Language Processing*, 2016.

[28] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[29] Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, pages 433–440. ACL, July 2006.

[30] Ofir Press and Lior Wolf. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*, 2016.

[31] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[32] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[33] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[34] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc., 2015.

[35] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

[36] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[37] Vinyals & Kaiser, Koo, Petrov, Sutskever, and Hinton. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, 2015.

[38] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[39] Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, and Wei Xu. Deep recurrent models with fast-forward connections for neural machine translation. *CoRR*, abs/1606.04199, 2016.

[40] Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. Fast and accurate shift-reduce constituent parsing. In *Proceedings of the 51st Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 434–443. ACL, August 2013.

[25] Mitchell P Marcus, Mary Ann Marcinkiewicz và Beatrice Santorini. Xây dựng một kho ngữ liệu tiếng Anh có chú thích lớn: The penn treebank. Ngôn ngữ học tính toán, 19(2):313–330, 1993.

[26] David McClosky, Eugene Charniak và Mark Johnson. Tự đào tạo hiệu quả để phân tích cú pháp. Trong Kỷ yếu Hội nghị Công nghệ Ngôn ngữ Con người của NAACL, Hội nghị Chính, trang 152–159. ACL, tháng 6 năm 2006.

[27] 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.

[28] 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.

[29] Slav Petrov, Leon Barrett, Romain Thibaux và Dan Klein. Học chú thích cây chính xác, nhỏ gọn và dễ hiểu. Trong Kỷ yếu của Hội nghị quốc tế lần thứ 21 về Ngôn ngữ học tính toán và Hội nghị thường niên lần thứ 44 của ACL, trang 433–440. ACL, tháng 7 năm 2006.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] Vinyals & Kaiser, Koo, Petrov, Sutskever và Hinton. Ngữ pháp như một ngoại ngữ. Trong những tiến bộ trong hệ thống xử lý thông tin thần kinh, 2015.

[38] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quốc V Lê, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 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.

[39] Jie Zhou, Ying Cao, Huguang 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.

[40] Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang và Jingbo Zhu. Phân tích thành phần giảm dịch chuyển nhanh và chính xác . Trong Kỷ yếu Hội nghị thường niên lần thứ 51 của ACL (Tập 1: Bài viết dài), trang 434–443. ACL, tháng 8 năm 2013.

Attention Visualizations

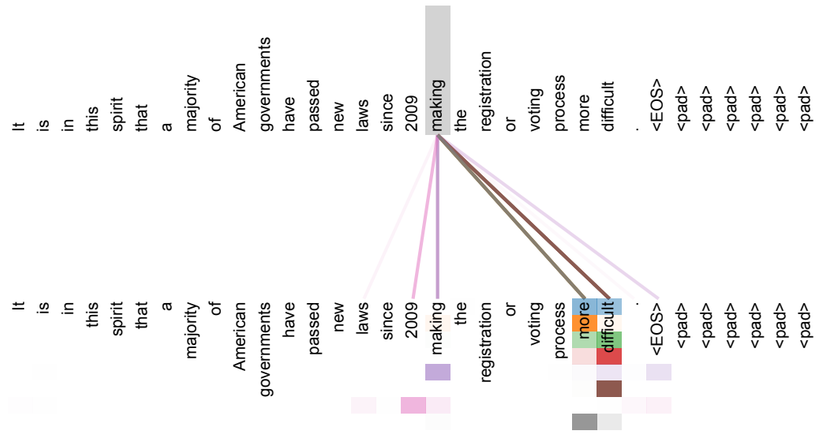
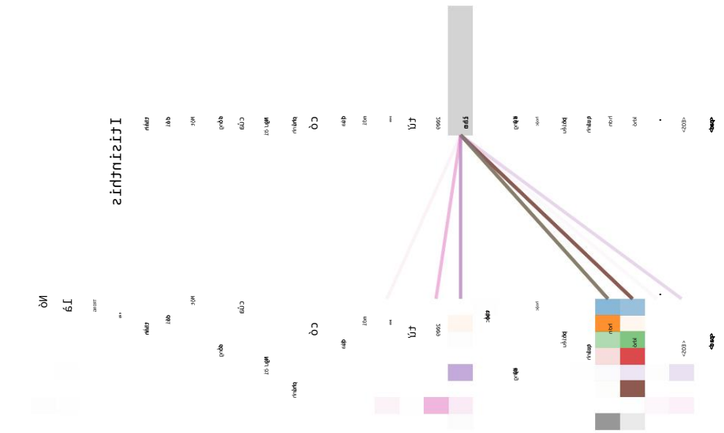


Figure 3: An example of the attention mechanism following long-distance dependencies in the encoder self-attention in layer 5 of 6. Many of the attention heads attend to a distant dependency of the verb ‘making’, completing the phrase ‘making...more difficult’. Attentions here shown only for the word ‘making’. Different colors represent different heads. Best viewed in color.

Lớp đầu vào-đầu vào5

Hình ảnh chú ý



Hình 3: Một ví dụ về cơ chế chú ý theo sau sự phụ thuộc khoảng cách xa trong khả năng tự chú ý của bộ mã hóa ở lớp 5 trên 6. Nhiều đầu chú ý chú ý đến sự phụ thuộc xa của động từ 'làm', hoàn thành cụm từ 'làm.. .khó hơn'. Sự chú ý ở đây chỉ đư ợc thể hiện đối với từ 'làm'. Màu sắc khác nhau đại diện cho những cái đầu khác nhau. Xem tốt nhất ở màu sắc.

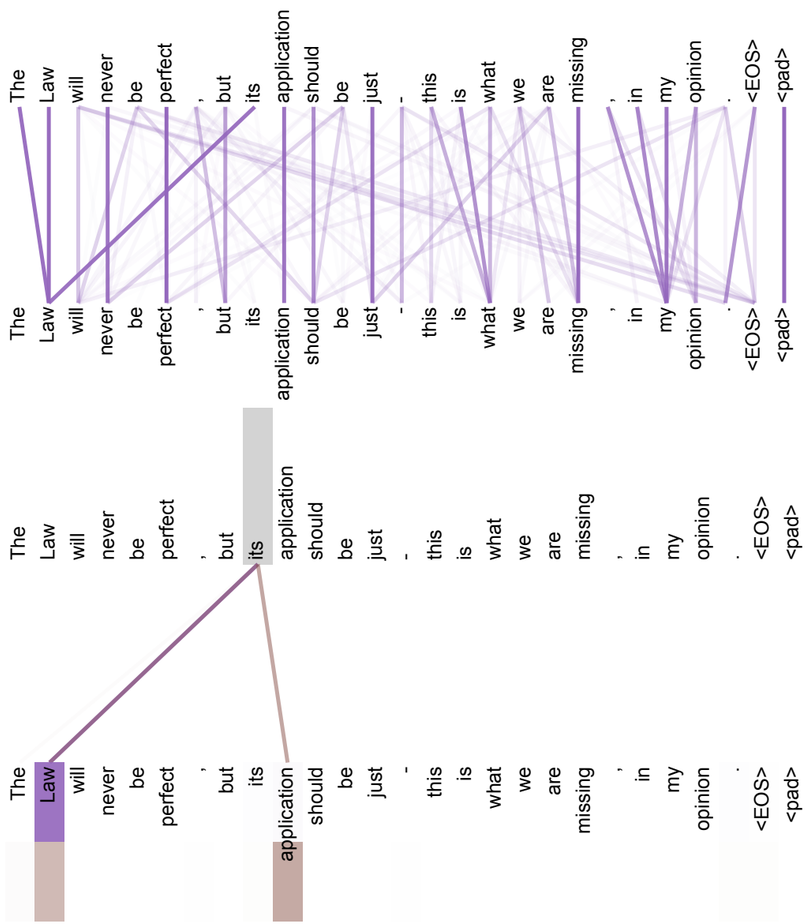
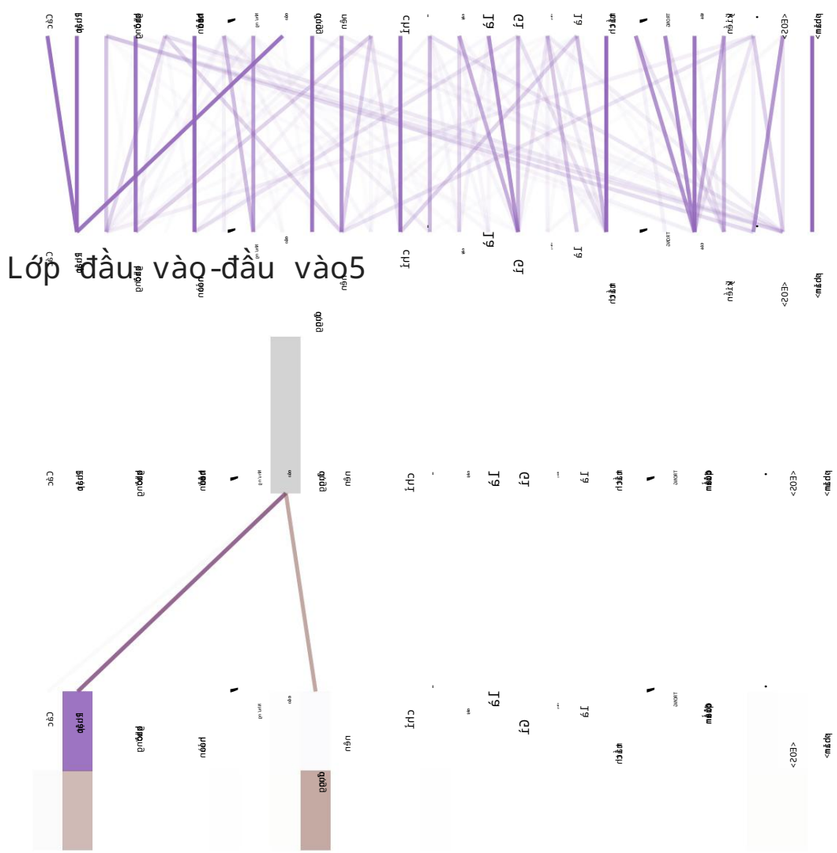


Figure 4: Two attention heads, also in layer 5 of 6, apparently involved in anaphora resolution. Top: Full attentions for head 5. Bottom: Isolated attentions from just the word ‘its’ for attention heads 5 and 6. Note that the attentions are very sharp for this word.

Lớp đầu vào-đầu vào5



Lớp đầu vào-đầu vào5

Hình 4: Hai đầu chú ý, cũng ở lớp 5 trên 6, dư ờng như có liên quan đến việc phân giải phép ẩn dụ. Trên cùng: Sự chú ý hoàn toàn dành cho đầu 5. Dư ới cùng: Sự chú ý tách biệt chỉ từ từ 'its' cho sự chú ý đầu 5 và 6. Lư u ý rằng sự chú ý rất rõ ràng đối với từ này.

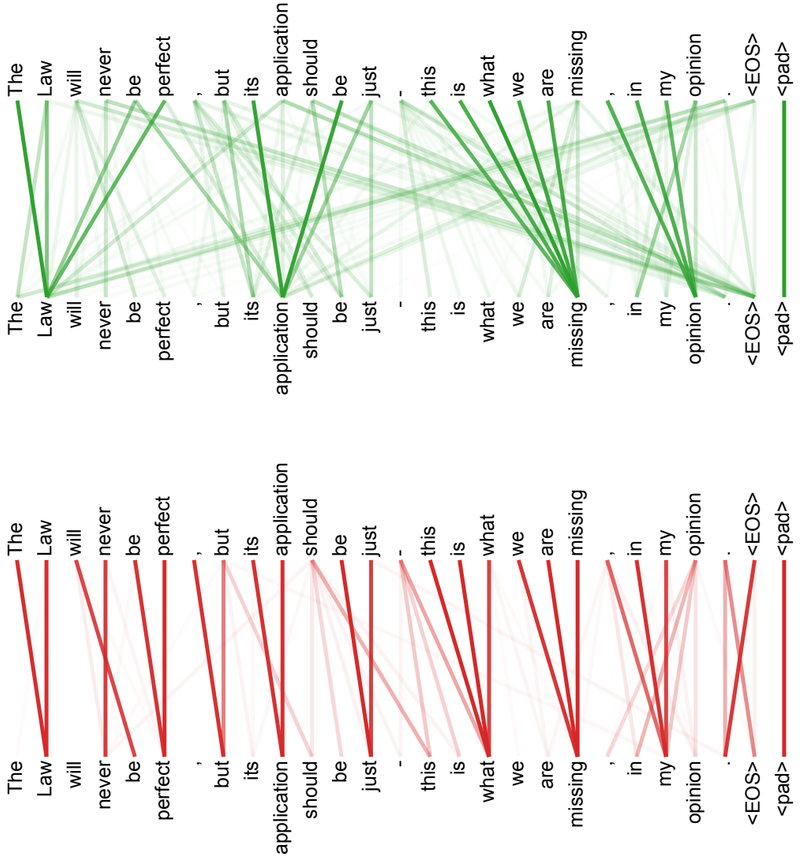
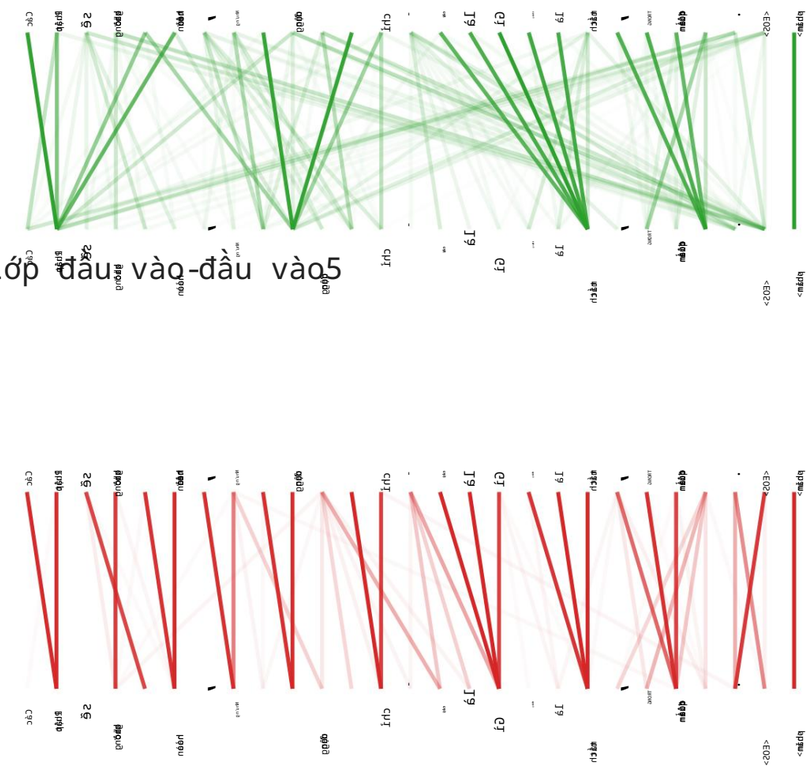


Figure 5: Many of the attention heads exhibit behaviour that seems related to the structure of the sentence. We give two such examples above, from two different heads from the encoder self-attention at layer 5 of 6. The heads clearly learned to perform different tasks.

Lớp đầu vào-đầu vào5



Lớp đầu vào-đầu vào5

Hình 5: Nhiều đầu chú ý thể hiện hành vi có vẻ liên quan đến cấu trúc của câu. Chúng tôi đư a ra hai ví dụ như vậy ở trên, từ hai đầu khác nhau từ bộ mã hóa tự chú ý ở lớp 5 trên 6. Các đầu rõ ràng đã học cách thực hiện các nhiệm vụ khác nhau.