Final Project Report

Môn học: Khai phá dữ liệu nâng cao

Học viên: Nguyễn Đức Trường

GVHD: TS. Cao Văn Chung

Lời tựa

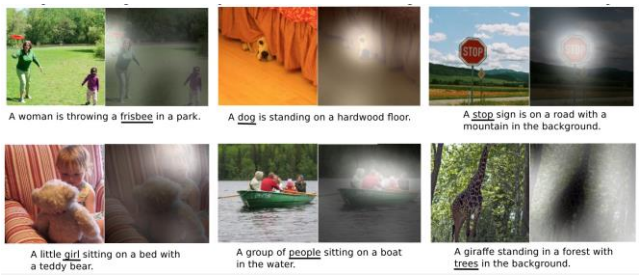
Chúng ta quan tâm đến việc máy tính có thể tự động mô tả hình ảnh theo ngôn ngữ của con người. Bài toán này có thể có thể có tác dụng trong việc gán nhãn tự động khi số gán nhãn thủ công sẽ rất tốn kém số lượng ảnh lớn, ngoài ra có thể ứng dụng trong việc hỗ trợ người khiếm thị….

Trong phạm vi bài trình bày này, chúng ta cùng tìm hiểu ứng dụng 1 số mạng deep learning trong bài toán này. Chúng tôi đã triển khai bài toán với 5 mục chính ***(R1) data prepossessing; (R2) Convolutional Neural Net-work (CNN) as an encoder (xử dụng transfer learning); (R3) attention mechanism; (R4) Recurrent Neural Network (RNN) as a decoder; (R5) Greedy Search để tìm được caption tốt nhất; (R6) sinh từ và đánh giá****. BLEU score được xử dụng trong bài toán này để đánh giá độ chính xác của caption.*

1. Giới thiệu

Bài toán này là sự kết hợp giữa hiểu hình ảnh, trích xuất đặc trưng cũng như dịch biểu diễn của hình ảnh thành ngôn ngữ tự nhiên. Với sự tiến bộ vượt bậc trong Mạng Neural, một số nhóm đã bắt đầu khám phá Mạng CNN và RNN để hoàn thành bài toán này và nhận thấy những kết quả rất tốt.

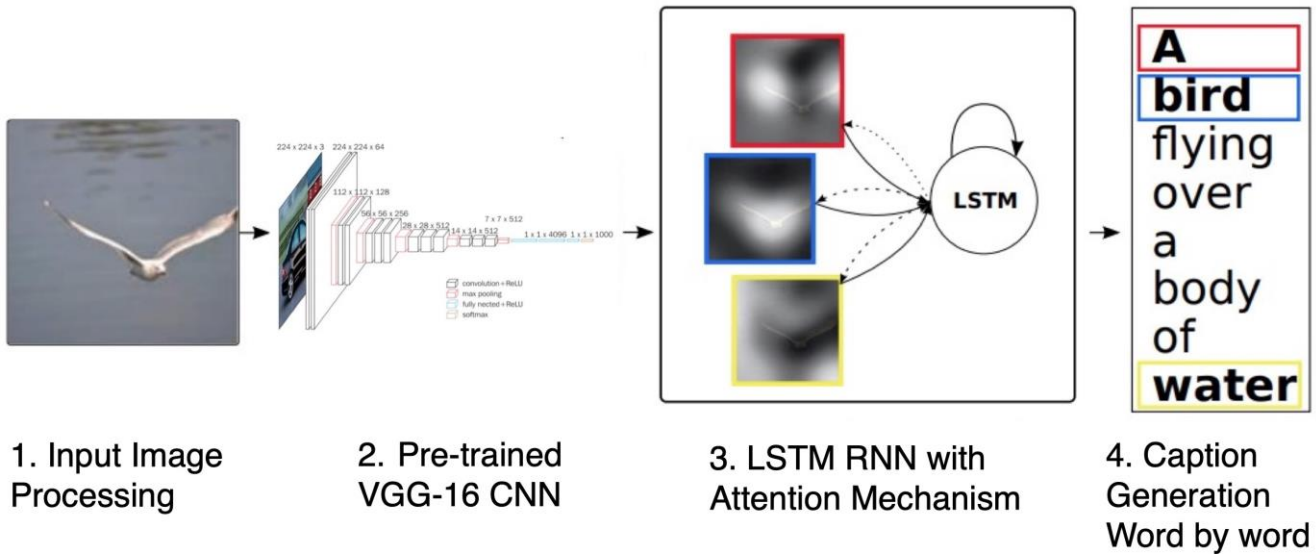
Một trong những paper phổ biến là: **Show and Tell: A Neural Image Caption Generator** và **Show, attend and tell: Neural image caption generator with visual attention**. Trong khi cả hai bài báo đều đề xuất sử dụng sự kết hợp của Mạng CNN và RNN để đạt được nhiệm vụ này, bài báo thứ hai được xây dựng dựa trên bài báo đầu tiên bằng cách thêm cơ chế attention. Như trong Hình 1, lớp attention có thể học được này cho phép mạng tập trung vào một vùng cụ thể của hình ảnh cho mỗi từ được tạo.



Hình 1. Hình ảnh từ paper Show, attend and tell visualization.

2. Phương pháp luận và kiến trúc

Bước đầu tiên là tiền xử lý dữ liệu đầu vào (cả hình ảnh và caption); 1 mạng pre-trained CNN như là lớp encoder để trích xuất đặc trưng (extract feature); 1 mạng LSTM-based Recurrent Neural Network là decoder để convert encoded features thành ngôn ngữ tự nhiên; Cơ chế Attention giúp cho phần decoder có thể tập chung vào 1 vùng cụ thể của input đầu vào để giúp cải thiện performance. Beam Search để tìm caption với khả năng cao nhất. Về chi tiết sẽ được trình bày dưới đây.



Hình 2. Show, attend and tell Architecture

2.1. Nguồn data

Đề tài đã sử dụng tập dữ liệu Flickr8k, trong đó mỗi hình ảnh được liên kết với 5 chú thích khác nhau mô tả các thực thể và sự kiện được mô tả trong hình ảnh đã được thu thập.

Flickr8k là một tập dữ liệu tốt cho thử nghiệm vì nó có kích thước nhỏ nên có thể dễ dàng training.

Cấu trúc tập dữ liệu của chúng tôi như sau:

- Flick8k /
    - Flick8k_Dataset /: - chứa ~8000 hình ảnh
    - Flick8k_Text /
        - Flickr8k.token.txt: - chứa id hình ảnh cùng với 5 chú thích

Tập dữ liệu cho tỷ lệ 8: 2 cho train và validate.

the white and brown dog is running over the surface of the snow .
a white and brown dog is running through a snow covered field .
a dog running through snow .
a dog is running in the snow
a brown and white dog is running through the snow .

man on skis looking at artwork for sale in the snow
a skier looks at framed pictures in the snow next to trees .
a person wearing skis looking at framed pictures set up in the snow .
a man skis past another man displaying paintings in the snow .
a man in a hat is displaying pictures next to a skier in a blue hat .

several climbers in a row are climbing the rock while the man in red watches and holds the line .
seven climbers are ascending a rock face whilst another man stands holding the rope .
a group of people climbing a rock while one man belays
a group of people are rock climbing on a rock climbing wall .
a collage of one person climbing a cliff .

large brown dog running away from the sprinkler in the grass .
a dog is playing with a hose .
a brown dog running on a lawn near a garden hose
a brown dog plays with the hose .
a brown dog chases the water from a sprinkler on a lawn .

a white dog running after a yellow ball
a white dog is ready to catch a yellow ball flying through the air .
a white dog is about to catch a yellow dog toy .
a white dog is about to catch a yellow ball in its mouth .
a dog prepares to catch a thrown object in a field with nearby cars .

2.2. Tiền xử lý

Dữ liệu đầu vào bao gồm hình ảnh và chú thích, do đó chúng ta cần xử lý trước cả hình ảnh ở định dạng thích hợp cho mạng CNN và chú thích văn bản thành định dạng thích hợp cho mạng RNN. Vì hệ thống tạo phụ đề hình ảnh của chúng tôi đang tận dụng mạng pretrained CNN (vgg-16), chúng ta cần chuyển đổi hình ảnh sang định dạng chính xác (3 ×H x W), với W và H ít nhất là 224 nên trong bài đã resize hình ảnh về kích thước 224x224.

2.3. Convolutional Neural Network (Encoder)

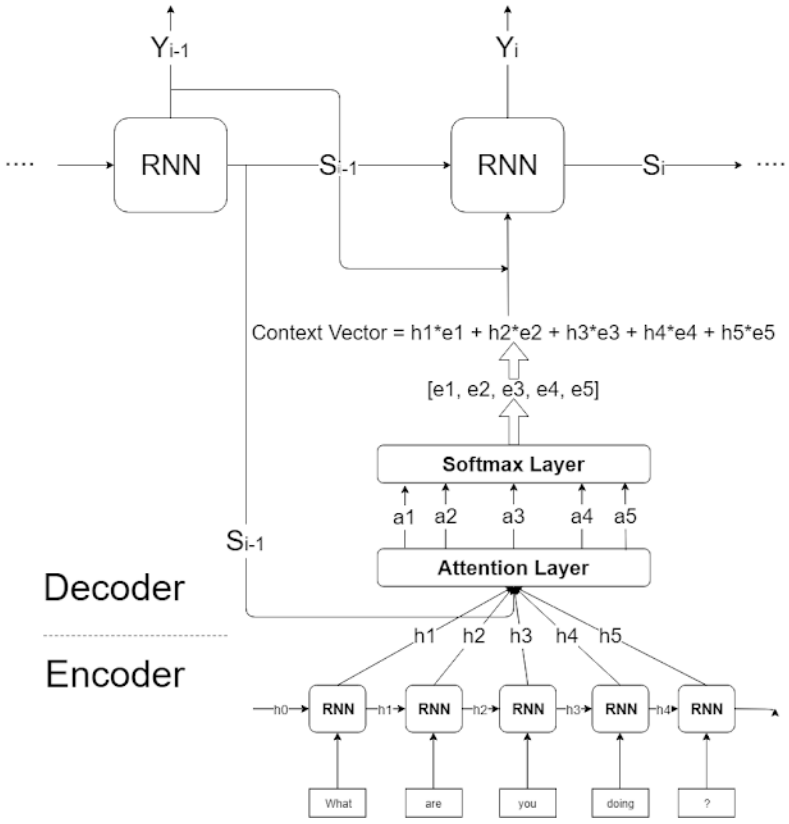
Bộ Encoder cần trích xuất các đặc điểm hình ảnh có nhiều kích thước khác nhau và mã hóa chúng thành không gian vectơ có thể được cung cấp cho RNN trong giai đoạn sau. VGG-16 và ResNet thường được đề xuất làm bộ mã hóa hình ảnh. Chúng tôi đã chọn sửa đổi mô hình pretrained VGG-16.
Trong bài toán này, CNN được sử dụng để encode các feature thay vì phân loại hình ảnh. Vì vậy, chúng tôi đã loại bỏ các lớp fully connected layers và max pool layers ở cuối mạng. Theo cấu trúc mới này, ma trận hình ảnh đầu vào có kích thước Nx3x224x 224 và đầu ra có kích thước Nx14 x14 x512.

2.4. Cơ chế Attention

Việc encode toàn bộ thông tin từ source vào 1 vector cố định khiến việc mô hình khi thực hiện trên các câu dài (long sentence) không thực sự tốt, mặc dù sử dụng LSTM (BiLSTM, GRU) để khắc phục điểm yếu của mạng RNN truyền thống với hiện tượng Vanishing Gradient, nhưng như thế có vẻ vẫn chưa đủ, đặc biệt đối với những câu dài hơn những câu trong training data. Từ đó, trong paper, tác giả Bahdanau đề xuất 1 cơ chế cho phép mô hình có thể chú trọng vào những phần quan trọng (word liên kết với word từ source đến target), và thay vì chỉ sử dụng context layer được tạo ra từ layer cuối cùng của Encoder, tác giả sử dụng tất cả các output của từng cell qua từng timestep, kết hợp với hidden state của từng cell để "tổng hợp" ra 1 context vector (attention vector) và dùng nó làm đầu vào cho từng cell trong Decoder.

Cách hoạt động của Attention

Trước khi đi vào những những công thức chi tiết về Attention, chúng ta cùng dạo qua chu trình hoạt động. Như đã nói ở trên, để hiện thực hóa ý tưởng về sự chú ý Attention, trong mô hình Sequence to Sequence thay vì chỉ giữ lại véc tơ trạng thái ẩn cuối cùng của Encoder, ta giữ lại toàn bộ các véc tơ trạng thái ẩn tại tất cả các thời điểm.



Hình 3: Mô hình attention

Như trên hình, ta giữ lại toàn bộ véc tơ trạng thái ẩn từ h1 đến h5 của Encoder. Đưa toàn bộ các véc tơ trạng thái ẩn qua một Attention Layer, Attention Layer này có thể đơn giản chỉ là một mạng nơ ron lan truyền tiến (Feed forward neural networks), nhận đầu vào là tất cả các véc tơ trạng thái ẩn của Encoder và véc tơ trạng thái ẩn tại thời điểm trước của Decoder (chính là S_{i-1} của RNN layer). Đầu ra sẽ là các hệ số "a1, a2, a3, a4, a5" tương ứng với các véc tơ trạng thái ẩn. Các hệ số này thể hiện mức độ chú ý của Decoder ở thời điểm hiện tại vào từng trạng thái ẩn. Đưa các hệ số này qua một lớp Softmax để chuẩn hóa thành các trọng số của sự chú ý (Attention weights), và đảm bảo các tính chất:

- Toàn bộ các trọng số nằm trong khoảng 0,1.
- Tổng các trọng số bằng 1.

Xét ví dụ trên hình 6, tại thời điểm "i" của Decoder giả sử các trọng số e1=0.1, e2=0.6, e3=0.15, e4=0.1, e5=0.05. Khi đó, thời điểm Decoder dự đoán Y_i phải dành sự chú ý nhiều hơn vào h2 vì e2 là lớn nhất. Các trọng số "e_j" có thể được coi là xác xuất tại thời điểm "i" Decoder chú ý nhiều hơn vào véc tơ trạng thái ẩn h_j (hay cũng là chú ý nhiều hơn vào phần tử đầu vào thứ j).

Context vector sẽ được tính bằng cách lấy tổng của các tích giữa véc tơ trạng thái ẩn và các véc tơ trọng số:

$$C_i = \sum_j h_j \cdot e_j$$

Lấy Context vector ghép nối trực tiếp với đầu ra của Decoder ở thời điểm trước (Y_{i-1}) và đưa vào làm đầu vào cho Decoder ở thời điểm hiện tại. Sau đó, Decoder sẽ tạo thành véc tơ đầu ra Y_i và véc tơ trạng thái ẩn S_i, cả hai đều được sử dụng trong thời điểm tiếp theo.

Chú ý: Ở thời điểm đầu tiên, khi chưa có S_{i-1} thì véc tơ trạng thái ẩn cuối cùng của Encoder có thể được dùng thay thế.

Chi tiết hơn vào các công thức toán:

Phần diễn giải ở trên dựa theo mô hình được đề xuất bởi Bahdanau[4] năm 2015, do đó phần công thức diễn giải tiếp theo cũng dựa trên bài báo này. Trong đó, RNN Encoder nhận đầu vào là x = {x1, . . . , xT } và tạo ra chuỗi véc tơ trạng thái ẩn h = {h1, . . . , hT }. Chuỗi véc tơ **h** có thể được coi là bộ nhớ (memory) của việc duyệt toàn bộ chuỗi **x**, nó còn là thể hiện của **x** trong một "không gian véc tơ ẩn" (latent space). Sau đó RNN Decoder tạo ra chuỗi đầu ra y = {y1, . . . , yU } dựa trên h theo cách hoạt động ở phần trên.

Trong quá trình tính y_i, Decoder sử dụng một hàm phi tuyến a(.) (Những hàm phi tuyến được nhắc đến trong bài đều là những hàm có thể học được và khả vi) để tính các trọng số $a_{i,j}$ tương ứng với mỗi h_j tại thời điểm *i* của Decoder. Hàm a(.) thông thường được xấp xỉ bằng một mạng nơ ron lan truyền tiến (Chính là Attention Layer trên hình 6) với hàm hàm kích hoạt là hàm *tanh*, tuy nhiên thì cũng có nhiều loại hàm khác được sử dụng cho những mục đích khác nhau. Những trọng số này sau đó được chuẩn hóa bởi một lớp softmax để tạo thành phân bố xác suất trên chuỗi véc tơ **h**[5].

$$a_{i,j} = a(s_{i-1}, h_j)$$
$$e_{i,j} = \exp(a_{i,j}) / \sum_{k=1}^{T} \exp(a_{i,k})$$

Trong đó $e_{i,j}$ là trọng số đã được chuẩn hóa tương ứng với h_j tại thời điểm i, và T là độ dài của chuỗi đầu vào. Context vector được tính dựa trên tổng của tích các h_j và trọng số $e_{i,j}$ tương ứng của nó theo công thức sau:

$$c_i = \sum_{j=1}^{T} e_{i,j} h_j$$

Cuối cùng Decoder tính s_i và sau đó là y_i dựa trên s_{i-1}, y_{i-1} và c_i:

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

$$y_i = g(s_i, c_i)$$

Trong đó f(.) được xấp xỉ bằng một mạng RNN, và g(.) là một hàm phi tuyến tạo đầu ra của Decoder từ các véc tơ trạng thái ẩn của mạng RNN

2.5. Recurrent Neural Network (Decoder)

Bộ decoder cần tạo phụ đề hình ảnh từng từ bằng cách sử dụng Mạng LSTM có thể tạo ra các từ một cách tuần tự. Đầu vào cho bộ giải mã là các vectơ đặc trưng hình ảnh được mã hóa từ CNN và chú thích hình ảnh được mã hóa được tạo ra trong giai đoạn tiền xử lý dữ liệu.

Bộ decoder bao gồm một mô-đun attention và một mô-đun LSTM và bốn lớp được kết nối đầy đủ do thư viện PyTorch cung cấp để khởi tạo các trạng thái của LSTMcell và từ điển từ.

k

2.6. Greedy Search và Beam Search

Trong các bài toán NLP như dịch máy (machine translation), tạo caption ảnh tự động (image caption generation), tóm tắt văn bản (text summarization), tổng hợp tiếng nói (auto speech recognition), …yêu cầu đầu ra của mô hình là chuỗi các từ có trong từ điển. Thường thì mỗi từ trong chuỗi từ mà mô hình của các bài toán như trên dự đoán (predict) sẽ đi kèm theo một phân phối xác suất tương ứng. Khi đó, bộ Decoder của mô hình sẽ dựa trên phân bố xác suất đó để tìm ra chuỗi từ phù hợp nhất. Tìm kiếm chuỗi từ phù hợp nhất yêu cầu chúng ta cần duyệt qua tất cả các chuỗi từ có thể có từ dự đoán của mô hình. Thường thì từ điển của chúng ta sẽ có kích thước rất lớn, với các bài toán về tiếng Việt thì khoảng trên dưới 30.000 từ khác nhau (bao gồm các từ tiếng anh phổ biến, tiếng Việt thuần thì ít hơn) hoặc nếu làm với dữ liệu tiếng anh thì kích thước bộ từ điển còn lớn hơn nhiều. Khi đó, không gian tìm kiếm của chúng ta là kích thước từ điển lũy thừa với độ dài của chuỗi.
Do không gian tìm kiếm là quá lớn. Trên thực tế, chúng ta thường dùng một thuật toán tìm kiếm heuristic để có được một kết quả tìm kiếm đủ tốt (good enough) cho mỗi dự đoán thay vì phải tìm kiếm toàn cục. Mỗi chuỗi từ sẽ được gán một số điểm dựa trên xác suất phân bố của chúng, thuật toán tìm kiếm sẽ dựa trên điểm số này để đánh giá các chuỗi từ này. Có 2 thuật toán tìm kiếm phổ biến giúp tìm ra chuỗi từ "đủ tốt" đó là tìm kiếm tham lam (greedy search) và beam search.

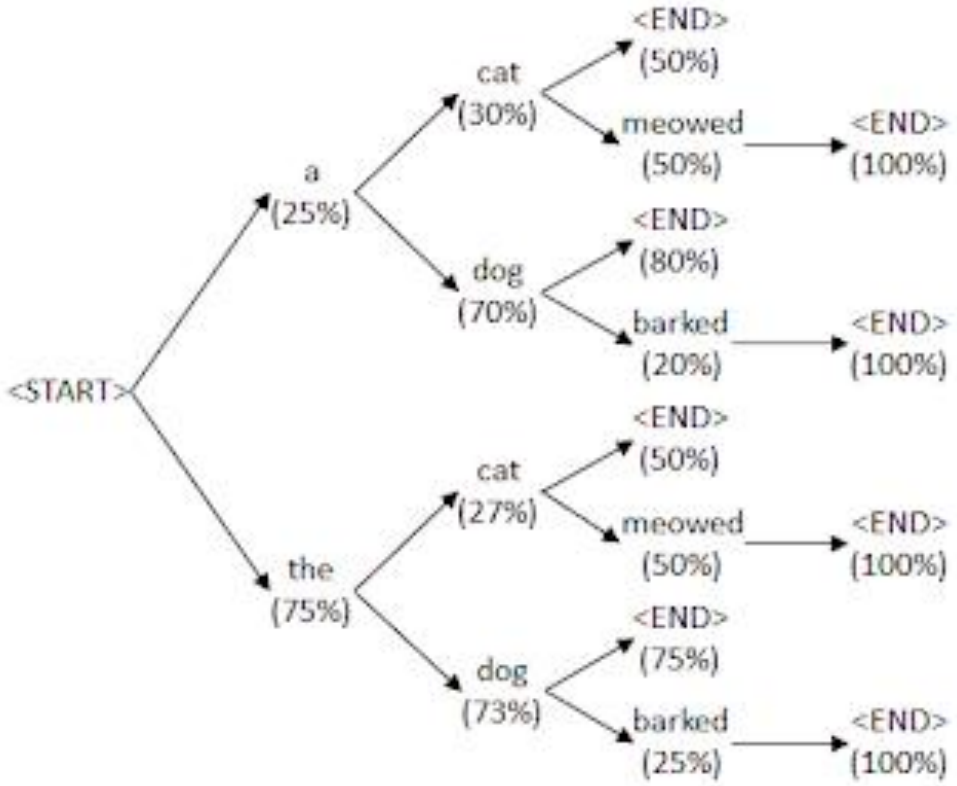
Greedy search vs Beam search

Trong phần này, mình sẽ cùng các bạn đi làm rõ ý tưởng của 2 thuật toán greedy search và beam search. Từ đó, bạn có thể thấy được ưu điểm, nhược điểm riêng của mỗi thuật toán cũng như lý do tại sao beam search hoạt động hiệu quả hơn.

Giải thuật tìm kiếm tham lam (greedy search) khởi đầu với chuỗi rỗng. Tại mỗi bước, nó thực hiện tìm kiếm toàn bộ trên không gian của bước đó và chỉ lấy duy nhất 1 kết quả có điểm số cao nhất và bỏ qua tất cả các kết quả khác. Sang bước tiếp theo, nó chỉ mở rộng tìm kiếm từ kết quả duy nhất trước đó.

Giải thuật toán kiếm beam search cũng khởi đầu với chuỗi rỗng. Tại mỗi bước, nó thực hiện tìm kiếm toàn bộ trên không gian của bước đó và lấy ra k kết quả có điểm số cao nhất thay vì chỉ lấy 1 kết quả cao nhất.

Hình ảnh dưới đây sẽ cho bạn thấy rõ nhất cách hoạt động của beam search với k =2.



3. Evaluation

3.1. Evaluation Metrics

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

thống dịch.

Việc đánh giá được thực hiện trên kết quả thống kê mức độ trùng khớp các n-grams (dãy ký tự gồm n từ hoặc ký tự) từ kho dữ liệu của kết quả dịch và kho các bản dịch tham khảo có chất lượng cao. Giải thuật của IBM đánh giá chất lượng của hệ thống dịch qua việc trùng khớp của các n-grams đồng thời nó cũng dựa trên cả việc so sánh độ dài của các bản dịch. Giá trị score đánh giá mức độ tương ứng giữa hai bản dịch và nó được

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

Công thức để tính điểm đánh giá BLEU score như sau:

N-Gram precision

$$p_n = \frac{\sum_{n\text{-gram} \in hyp} count_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in hyp} count(n\text{-gram})}$$

Bounded above by highest count of n-gram in any reference sentence

brevity penalty

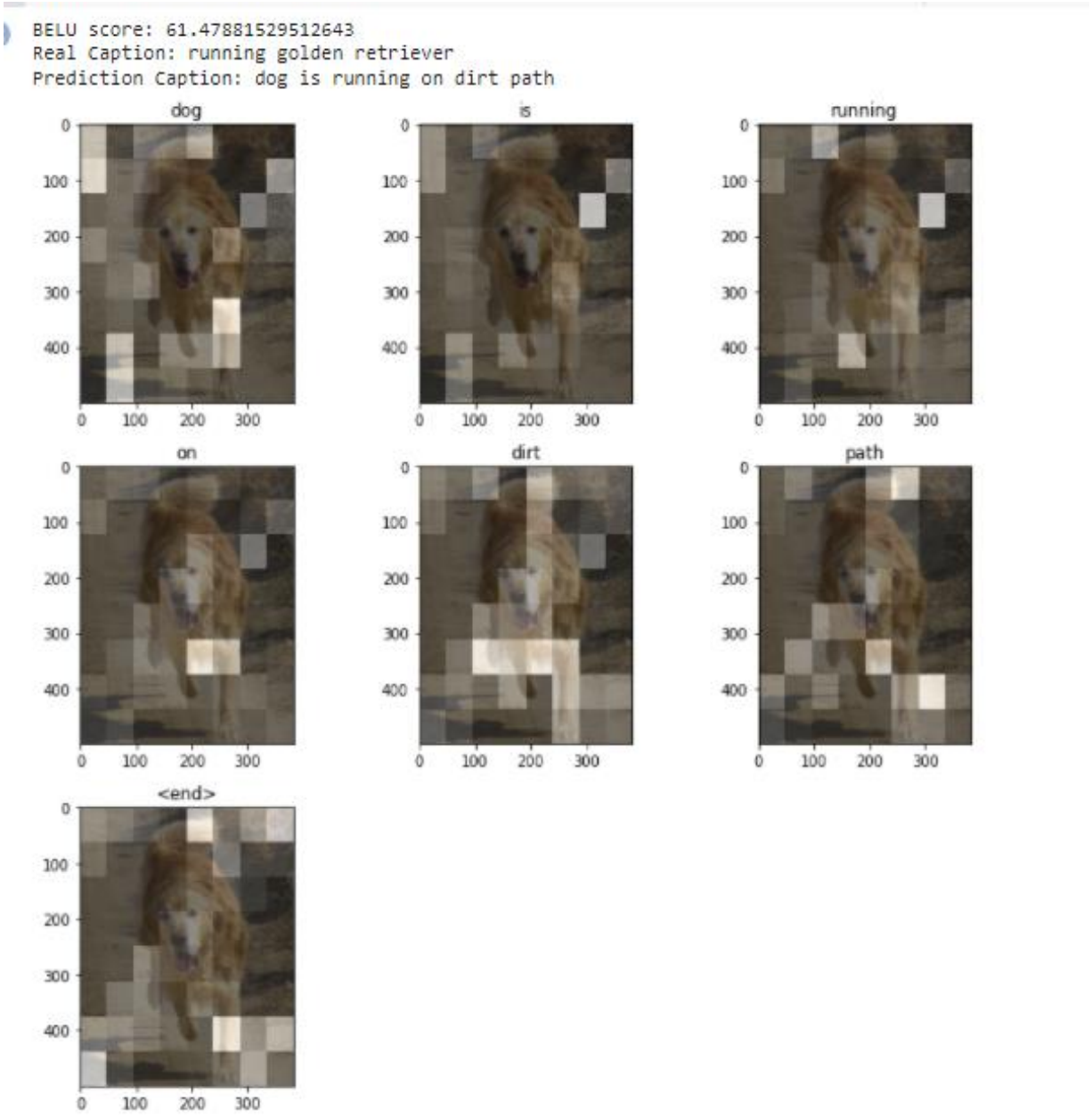
$$B = \begin{cases} e^{(1-|ref|/|hyp|)} & \text{if } |ref| > |hyp| \\ 1 & \text{otherwise} \end{cases}$$

Bleu score: brevity penalty, geometric mean of N-Gram precisions

$$Bleu = B \cdot \exp\left[\frac{1}{N}\sum_{n=1}^{N} p_n\right]$$

3.2. Attention Mechanism Visualization

Cơ chế attention cho phép chúng ta hiểu phần nào củ hình ảnh đang được tập trung khi tạo một từ cụ thể, điều này rất cần thiết để cải thiện khả năng hiểu cảnh của mô hình. Chúng tôi đã thực hiện



4. Kết luận

Tự động chú thích hình ảnh còn lâu mới hoàn thiện và có rất nhiều dự án nghiên cứu đang thực hiện nhằm mục đích trích xuất đặc điểm hình ảnh chính xác hơn và tạo câu tốt hơn về mặt ngữ nghĩa. Chúng tôi đã hoàn thành những gì chúng tôi đã đề cập trong phần giới thiệu, nhưng sử dụng tập dữ liệu nhỏ hơn (Flickr8k) do khả năng tính toán hạn chế. Có thể có những cải tiến tiềm năng thêm sau:

Trước hết, chúng tôi đã trực tiếp sử dụng mạng CNN VGG16. Bằng cách thử nghiệm với các mạng Pretrained CNN cho phép tinh chỉnh khác, chúng tôi hy vọng sẽ đạt được điểm cao hơn một chút.

Một cải tiến tiềm năng khác là kết hợp của Flickr8k, Flickr30k và MSCOCO. Nói chung, mạng càng có nhiều tập dữ liệu đào tạo đa dạng thì kết quả đầu ra càng chính xác.

5. Tài liệu tham khảo

- https://github.com/varun-bhaseen/Image-caption-generation-using-attention-model
- Local Attention : https://arxiv.org/pdf/1502.03044.pdf
- Global Attention : https://arxiv.org/pdf/1508.04025.pdf
- https://www.appliedaicourse.com/lecture/11/applied-machine-learning-online-course/4150/attention-models-in-deep-learning/8/module-8-neural-networks-computer-vision-and-deep-learning
- Tensorflow Blog: https://www.tensorflow.org/tutorials/text/image_captioning
- https://machinelearningmastery.com/calculate-bleu-score-for-text-python/
- https://towardsdatascience.com/intuitive-understanding-of-attention-mechanism-in-deep-learning-6c9482aecf4f
- Neural Machine Translation(Research Paper):https://arxiv.org/pdf/1409.0473.pdf