Đề tài: Ứng dụng cơ chế attention cho bài toán tự động mô tả hình

Học viên: Nguyễn Đức Trường

GVHD: TS. Cao Văn Chung

Hà Nội 2022

Mục lục

GIỚI THIỆU

- *Chúng ta quan tâm đến việc máy tính có thể tự động mô tả hình ảnh theo ngôn ngữ của con người. Để hiểu rõ hơn cơ chế của nó trong Computer vision,*

- *Bài toán sinh mô tả cho ảnh có thể có thể có tác dụng trong việc gán nhãn tự động khi số gán nhãn thủ công sẽ rất tiêu tốn khi 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 tôi 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.*

Bài toán mô tả hình ảnh

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing wi lego toy."

Nội dung cần có được

(R1) data prepossessing;

(R2) Convolutional Neural Net-work (CNN) as an encoder (xử dụng transfer learning);

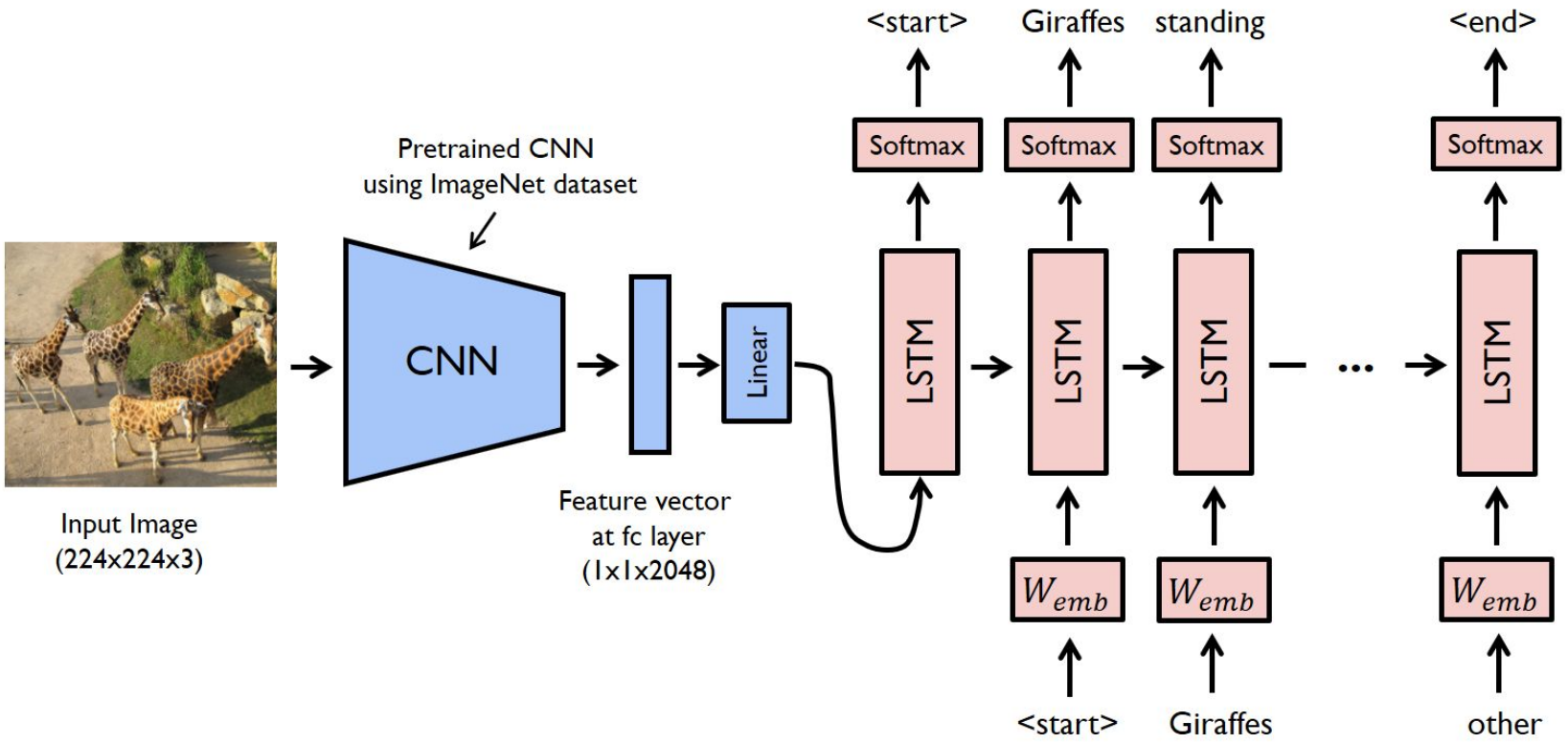
(R3) attention mechanism;

(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

A classic image captioning model



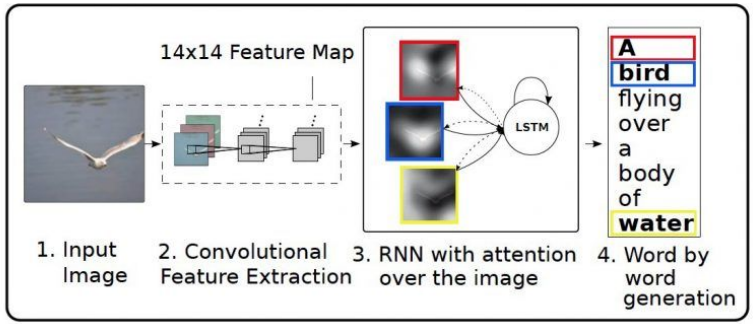
Issues with Traditional Methods of Image Captioning:

Mô hình CNN cần lưu trữ thông tin toàn bộ bức ảnh lại thành một vector duy nhất, rồi sau đó mô hình encoder sẽ phát sinh câu mô tả dựa theo thông tin lưu trong vector này, tuy nhiên, vector này có thể không chứa đủ thông tin để phát sinh cho toàn bộ câu mô tả.

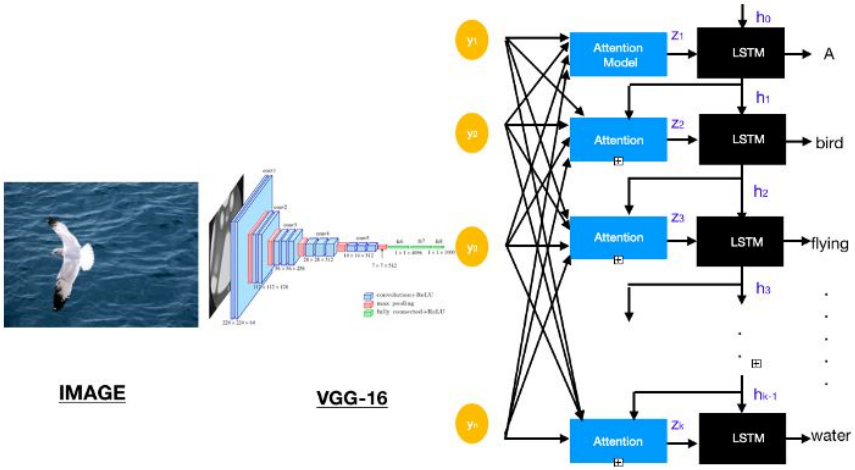
Để giải quyết vấn đề này, chúng ta sẽ sử dụng cơ chế attention, cho phép mô hình có thể chú ý vào từng phần của câu hoặc bức ảnh một cách rõ ràng. Từ đó,thông tin không cần phải nén vào một vector biểu diễn duy nhất. Ngoài ra, cơ chế attention cho phép mình có thể hiểu được những từ hay phần ảnh nào quyết định đến kết quả hiện tại.

Attention Mechanism Concept

- Chia hình ảnh thành n phần
- Dùng mạng Convolutional Neural Network (CNN) để trích xuất đặc trưng của mỗi phần thành h1,…, hn.
- Khi sinh 1 từ mới, cơ chế attention chỉ tập chung vào vùng ảnh có liên quan

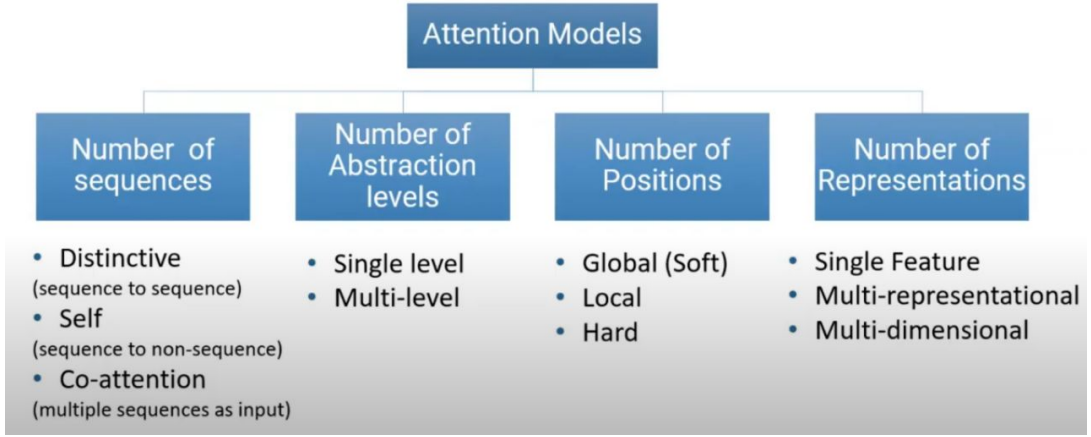


Concept of Attention Mechanism:



Điều gì đang xảy ra khi chúng ta muốn dự đoán từ mới của chú thích? Nếu chúng ta đã dự đoán từ i, hidden state của LSTM là hi. Chúng ta chọn phần «« relevant »» của hình ảnh bằng cách sử dụng chữ hi làm context. Sau đó, output của mô hình attention zi, là phần biểu diễn của hình ảnh được lọc sao cho chỉ còn lại các phần có liên quan của hình ảnh, được sử dụng làm đầu vào cho LSTM. Sau đó, LSTM dự đoán một từ mới và trả về trạng thái ẩn mới hi + 1.

Taxonomy of Attention



1. Soft Attention

2. Hard Attention

3. Multihead Attention

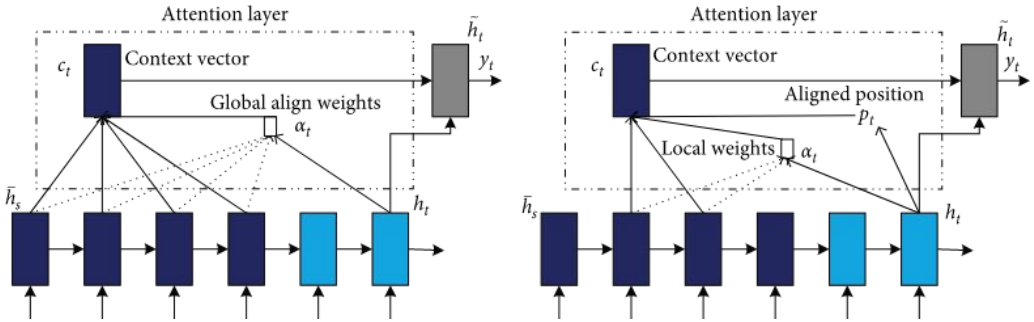
4. Scaled Dot-Product Attention

5. Global Attention (Luong's Attention):

Attention is placed on all source positions.

6. Local Attention (Bahdanau Attention):

Attention is placed only on a few source positions.



Reference : https://github.com/spro/practical-pytorch/blob/master/seq2seq-translation/seq2seq-translation.ipynb

Inference

- **Greedy search:** ● Maximum Likelihood Estimation (MLE) i.e. ● Selecting that word which is most likely according to the model for the given input. ● It’s also called as Greedy Search, as we greedily select the word with maximum probability.
- **Beam search:**

  Taking top k predictions ● Feed them again in the model ● sort them using the probabilities returned by the model. ● So, the list will always contain the top k predictions. ● Taking the one with the highest probability ● Going through it till we encounter or reach the maximum caption length.

Đánh giá mô hình

In the evaluation of sentence generation results,

BLEU

METEOR

ROUGE

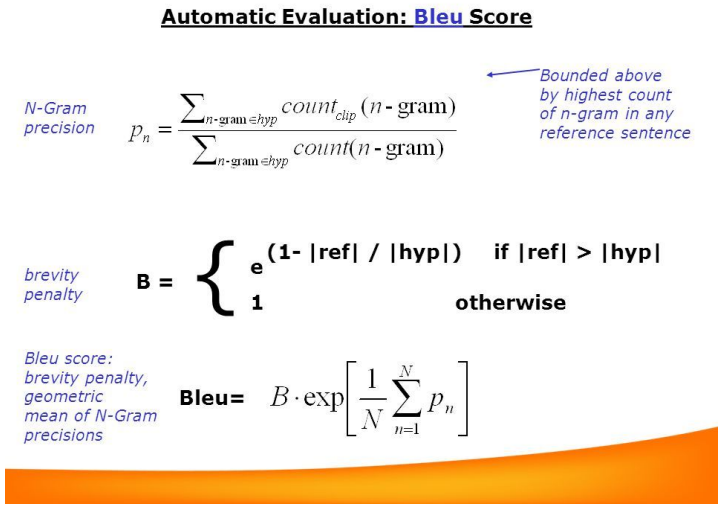
CIDEr

SPICE

BLEU score- (Bilingual Evaluation Understudy)

BLEU score định nghĩa thế nào là 1 mô tả tốt cho ảnh? tương tự như việc đánh giá độ chính xác của một Machine Translation. Nó thể hiện trong hai yếu tố:

- Adequacy: mô tả đầy đủ thông tin trong ảnh
- Fluency: đúng ngữ pháp, cấu trúc câu.




```
from torchtext.data.metrics import bleu_score
from nltk.translate.bleu_score
import sentence_bleu
```

Ứng dụng

Data Sets

Flickr8k

- 8000 images, each annotated with 5 sentences via AMT
- 1000 for validation, testing

Flickr 30k

- 30k images
- 1000 validation, 1000 testing

MSCOCO

- 123,000 images
- 5000 for validation, testing

Ứng dụng

1. Flickr8k_Dataset: Chứa tổng cộng 8092 hình ảnh ở định dạng JPEG với các hình dạng và kích thước khác nhau. Trong đó 6000 ảnh được dùng để train, 1000 ảnh để test và 1000 ảnh để validate
2. Flickr8k_text : gồm 3 bộ train_set ,test_set, và Flickr8k.token.txt chứa các từ đơn lẻ. mỗi ảnh có 5 CAPTION

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

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

the white and brown dog is running over the surface of the snow

white and brown dog is running through snow covered field

dog running through snow

dog is running in the snow

brown and white dog is running through the snow

man on skis looking at artwork for sale in the snow

skier looks at framed pictures in the snow next to trees

person wearing skis looking at framed pictures set up in the snow

man skis past another man displaying paintings in the snow

man in hat is displaying pictures next to skier in blue hat

tiền xử lý caption

- bỏ dấu

- bỏ chữ số

- bỏ các ký tự đơn

- thêm start và end vào để decoder biết điểm bắt đầu và kết thúc câu

```
    text_no_punctuation = text_original.translate(string.punctuation)
    return(text_no_punctuation)

def remove_single_character(text):
    text_len_more_than1 = ""
    for word in text.split():
        if len(word) > 1:
            text_len_more_than1 += " " + word
    return(text_len_more_than1)

def remove_numeric(text):
    text_no_numeric = ""
    for word in text.split():
        isalpha = word.isalpha()
        if isalpha:
            text_no_numeric += " " + word
    return(text_no_numeric)

def text_clean(text_original):
    text = remove_punctuation(text_original)
    text = remove_single_character(text)
    text = remove_numeric(text)
    return(text)

for i, caption in enumerate(data.caption.values):
    newcaption = text_clean(caption)
    data["caption"].iloc[i] = newcaption
```

```
PATH = "/content/drive/MyDrive/ML/Flicker8k_Dataset/"
all_captions = []
for caption  in data["caption"].astype(str):
    caption = '<start> ' + caption + ' <end>'
    all_captions.append(caption)

all_captions[:10]
```

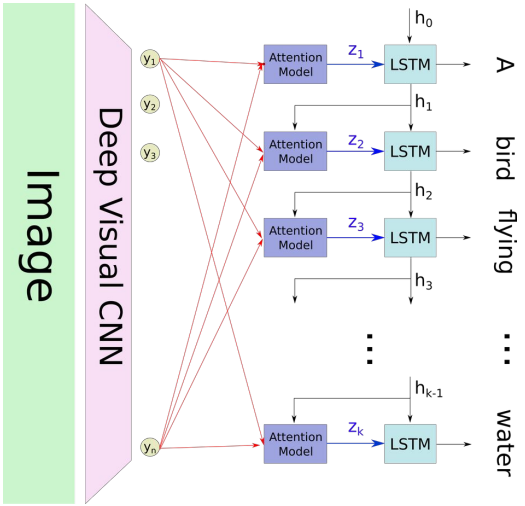
```
['<start>  child in pink dress is climbing up set of stairs in an entry way <end>',
 '<start>  girl going into wooden building <end>',
 '<start>  little girl climbing into wooden playhouse <end>',
 '<start>  little girl climbing the stairs to her playhouse <end>',
 '<start>  little girl in pink dress going into wooden cabin <end>',
 '<start>  black dog and spotted dog are fighting <end>',
 '<start>  black dog and dog playing with each other on the road <end>',
 '<start>  black dog and white dog with brown spots are staring at each other in the street <end>',
 '<start>  two dogs of different breeds looking at each other on the road <end>',
 '<start>  two dogs on pavement moving toward each other <end>']
```

```
top_k = 5000
tokenizer = tf.keras.preprocessing.text.Tokenizer(num_words=top_k,
                                                  oov_token="<unk>",
                                                  filters='!"#$%&()*+.,-/:;=?@[\]^_`{|}~ ')

tokenizer.fit_on_texts(train_captions)
train_seqs = tokenizer.texts_to_sequences(train_captions)
tokenizer.word_index['<pad>'] = 0
tokenizer.index_word[0] = '<pad>'

train_seqs = tokenizer.texts_to_sequences(train_captions)
cap_vector = tf.keras.preprocessing.sequence.pad_sequences(train_seqs, padding='post')
```

- tokenizer từ vựng
- dùng padding và <unk> cho newword
- giới hạn dung lượng từ vựng ở top 5000 để dễ tính toán

Xây dựng mô hình

- Number of epochs: 20
- Batch size: 64
- Loss function: `SparseCategoricalCrossentropy`
- Optimizer: Adam
- Metrics: BLEU (more below)
- Greedy search



- Trong bài dùng mô hình Image Model (VGG-16) pre-trained để làm phần encoder trích xuất đặc trưng

```python
def load_image(image_path):
    img = tf.io.read_file(image_path)
    img = tf.image.decode_jpeg(img, channels=3)
    img = tf.image.resize(img, (224, 224))
    img = preprocess_input(img)
    return img, image_path

image_model = tf.keras.applications.VGG16(include_top=False, weights='imagenet')
new_input = image_model.input
hidden_layer = image_model.layers[-1].output
image_features_extract_model = tf.keras.Model(new_input, hidden_layer)

image_features_extract_model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/v
58892288/58889256 [==============================] - 1s 0us/step
58900480/58889256 [==============================] - 1s 0us/step
```

- VGG16 model đã được train cho bài toán phân loại với bộ ImageNet data-set.
- Với bài toán này, ta loại đi layer fully connected ở cuối.

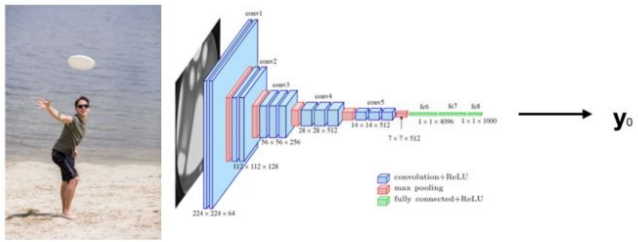


IMAGE **VGG-16 Encoder**

Defining Optimizer, and Loss Function

```
optimizer = tf.keras.optimizers.Adam()
loss_object = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True, reduction='none')

def loss_function(real, pred):
    mask = tf.math.logical_not(tf.math.equal(real, 0))
    loss_ = loss_object(real, pred)
    mask = tf.cast(mask, dtype=loss_.dtype)
    loss_ *= mask

    return tf.reduce_mean(loss_)
```

CHECK POINT SAVING

```
checkpoint_path_ckpt = "/content/drive/MyDrive/ML/checkpoint/train"
ckpt = tf.train.Checkpoint(encoder=encoder,
                           decoder=decoder,
                           optimizer = optimizer)
ckpt_manager = tf.train.CheckpointManager(ckpt, checkpoint_path_ckpt, max_to_keep=5)
```

Start checkpointing from the checkpoint last saved

```
[45] start_epoch = 0
     if ckpt_manager.latest_checkpoint:
       start_epoch = int(ckpt_manager.latest_checkpoint.split('-')[-1])
```

```
[ ]  optimizer = tf.keras.optimizers.Adam()
     loss_object = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True, reduction='none')

     def loss_function(real, pred):
         mask = tf.math.logical_not(tf.math.equal(real, 0))
         loss_ = loss_object(real, pred)
         mask = tf.cast(mask, dtype=loss_.dtype)
         loss_ *= mask

         return tf.reduce_mean(loss_)
```

CHECK POINT SAVING để load sau này

```
     checkpoint_path_ckpt = "/content/drive/MyDrive/ML/checkpoint/train"
     ckpt = tf.train.Checkpoint(encoder=encoder,
                                decoder=decoder,
                                optimizer = optimizer)
     ckpt_manager = tf.train.CheckpointManager(ckpt, checkpoint_path_ckpt, max_to_keep=5)
```

Start checkpointing from the checkpoint last saved

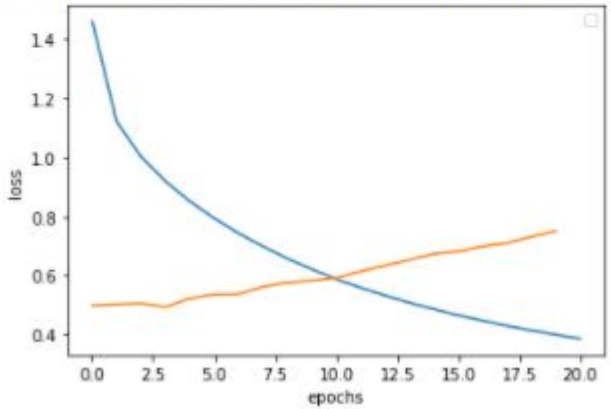
```
[ ]  start_epoch = 0
     if ckpt_manager.latest_checkpoint:
       start_epoch = int(ckpt_manager.latest_checkpoint.split('-')[-1])
```

Kết quả

BLEU score for all val image:
0.3906584935126492

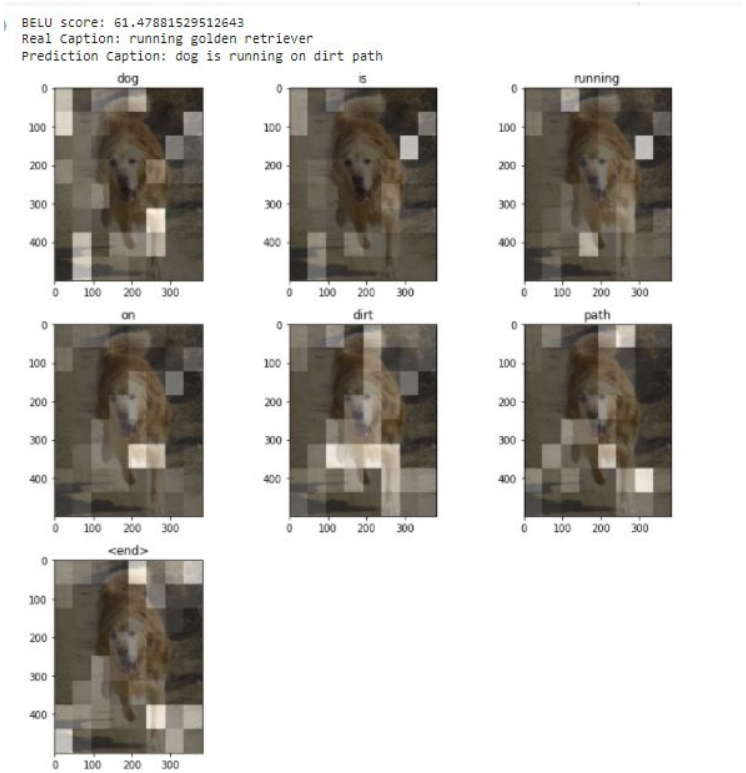
```
for label in [loss_plot,test_loss_plot]:
    plt.plot(label)
plt.legend()
plt.xlabel("epochs")
plt.ylabel("loss")
plt.show()
```

```
No handles with labels found to put in legend.
```



Visualize Attention

Nhận xét: Mô hình cho dự đoán Caption tương đối tốt, nhiều trường hợp còn miêu tả tốt hơn cả caption thực tế.



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
- 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.

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