Improving a Deep Learning model for Image Captioning

Ankur Kunder
Department of Computer Science and Engineering, Texas A&M University,
College Station, Texas, USA
a.kunder@tamu.edu

1. Abstract

Image Captioning is a method to generate description for images. It is easy for humans to describe an image quite efficiently, but it is quite difficult for machines. For this, the algorithm/model needs to detect the objects in the image, their inherent relations to one another, and then generate a description that is semantically and syntactically correct. Solving this problem is particularly useful for Content Based Image Retrieval. A large proportion of data on Social Media websites are images, therefore in order to recommend content or to help with captions/hashtags, it becomes a problem of interest to automatically generate captions based on the contents of an image. In this project, Visual Attention method is used to improve a Deep Learning Model for Image Caption Generation.

2. Background

As mentioned earlier, for image captioning, objects should be detected first and then a description based on their relations needs to be generated. As a result, Deep Learning Models for Image Captioning have two modules, an Image Encoder and a Language Model. The Image Encoder consists of a Deep Neural Network that encodes the image information, which is then fed to a Language model which generates the parts of captions sequentially. The models that have shown good accuracy use a VGGNet[18] as the Image Encoder and an RNN/LSTM as the Language Model[9].

2.1. Image Encoder Models

Object detection models are used to encode the images. A layer before the last one is considered a good source of image features and therefore, used as the encoded feature vector. VGGNets[18] are an improvement over AlexNet[12], and are a popular choice for extracting image features. They take a 224x224 pixel, center-cropped images as input. The use of small size 3x3 filters with stride 1, in contrast to 11x11 size filters with stride 4 of AlexNet, allows them to have more number of CNN layers, which leads to better performance. VGG16 uses 1x1 convolution layers to increase non-linearity of the model. There are many configurations for a VGGNet, they are shown in the attached figure taken from the original paper[SITE THE VGG PAPER]. They also have 3 fully-connected layers in the end, with 1000 output units for the ImageNet classification challenge. Generally, one of the layers before the last layer is taken as a feature vector for training Deep Learning models that have images as input. The VGGNet layers till the chosen layer may or may not be retrained/fine-tuned along with the rest of the model on another task. This depends on various factors such as presence of trainable parameters after the extracted image features or the availability of computing resources. In both of these cases, the layers are not retrained, and the new model takes the extracted image features as the input.

2.2. Language Model

A language model predicts the occurrence of a word given a sequence of words that come from a text. Since, RNNs can learn the relations between elements in sequential data, they were a popular choice for language modeling. RNNs can also handle variable length inputs, as they use the previous hidden state of the network to compute the output of the current state. But, due to the problem of gradient decay/explosion in RNNs, they are unable to learn long term dependencies in between the words. As a result, LSTMs[7] are used to avoid this problem as they have direct connections from one cell to another, which allows the gradient to flow without interacting with the output. The LSTM controls the learning process through gates known as forget gate, input gate, and output gate. The equations are given below, along with Figure 3 for reference.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

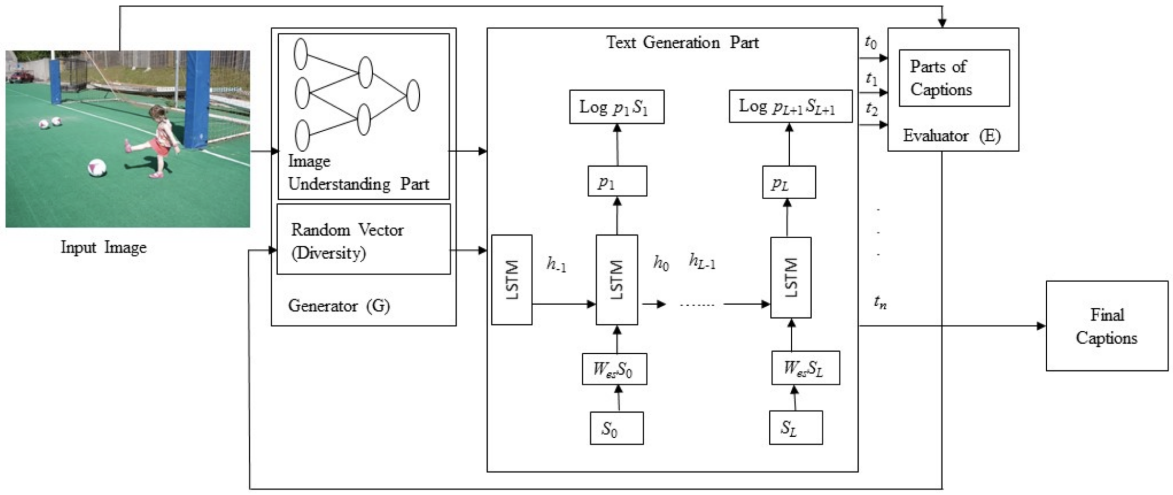

Figure 1. A Block Diagram by Hossain et al. [9] for Deep Learning Based Image Captioning.

$$h_t = o_t \cdot tanh(C_t)$$

Where f is the forget gate, i is the input gate, o is the output gate, h is the hidden state, x is the input word vector, and C is the cell state. The σ and $tanh$ are used to control which input contributes, and by how much, to the output and the next state.

Write about what language modeling does, variable length sequences, what are the benefits of using an LSTM over an RNN. Mention the basic equations of LSTM gates. Insert an Image. Talk about major methods that have used LSTMs as a language model.

2.3. Related Work

There have been many approaches based on Deep Learning models for Image Captioning. Apart from the Vanilla CNN as Image Encoder and LSTM as Language Model, following are some of the models that resulted in better quality of captions. Johnson et al. [11] proposed Region Proposal using a CNN and then generating captions using a language model. This is called Dense Captioning. Therefore, instead of generating caption for the entire scene at once, captions are generated for different regions.

In order to improve the Language Model part, Wang et al. [20] proposed a deep bidirectional LSTM-based method for image captioning. This model could use both the past and future context for language generation. This again supports the observation that more information leads to better accuracy, and some kind of feedback at each time step could improve the performance of the language model.

In another approach[6], GANs are used to translate images to captions by use of a Generator and Discriminator Network. The Generator learns it's loss value from the Discriminator instead of the labeled data. The Discriminator, which has the true data, only informs the Generator whether the generated captions are real or fake. This helps them to generate captions similar to real captions.

In Attention Based Image Captioning, different regions of the image are referred at each time step of the language generation model, based on the words it previously generated, to generate the next word. This allows dynamic update of captions based on the region of the network is paying attention to in the current time step. Many models have been proposed based on how they factor in the attention to a particular region [23][21][14][10][4].

It still remains an area of interest to add attention to the models for generating better captions. Taking the last fully connected layer of the CNN as input to the Language Model gives inferior quality of results, because the receptive field region of this layer is quite large, which leads to poor spatial attention for the image.

3. Method Description

3.1. Preliminary Approach

To learn about text generation, a model based on Jason Brownlee's approach[3] was made, which tried to predict a word, given the image and some text description related to it. The model takes the features from the last layer of VGG16 as one of the inputs. It also processes the sentences using an LSTM layer to provide its encoded

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 2. A table by Simonyan and Zisserman[18] for VGG Configurations. The depth increases from left to right. Naming follows the format of conv¡filter-size¿-¡number-of-outputs. The ReLU are not shown for brevity. VGG16 and VGG19 have been used in this project and are marked by a red box.

vector representation as the second input.

For this, the captions were preprocessed first, and each word was converted into a number based on the vocabulary size and it's frequency. To do this, the captions were first converted to lowercase, 'startseq' and 'endseq' tokens appended to the front and back of the captions, and symbols like #$% etc. were removed to simplify vocabulary. After this, a tokenizer was learnt after processing all the available captions in the training data. This tokenizer then provided the dictionaries for word-to-index and index-to-word conversion. This was done using keras and Tensorflow APIs. The caption was then converted to a sequence of numbers, which were then passed to an Embedding layer that tried to learn a vector representation for each word, such that in the new space similar words had similar vector representations. These vector representation of words were then used as input for the LSTM.

The VGG16 layers, that are used to extract the features

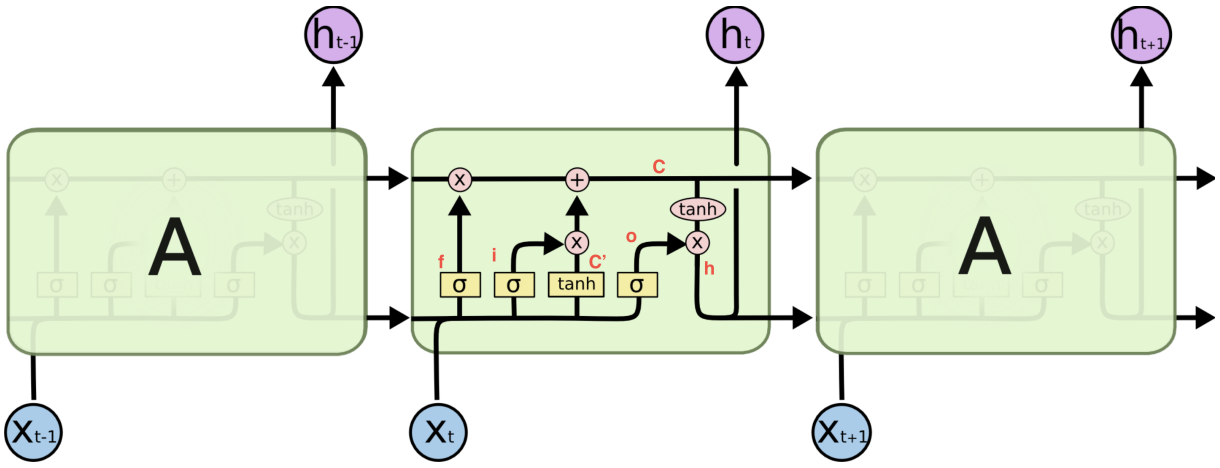

Figure 3. An LSTM Network visualized by Christopher Olah[15].

for the image, are not fine-tuned with the rest of the model. This is done in order to make the model realizable under constrained computing resources. Therefore, the image features are extracted beforehand and used with the captions for training.

Then, these two inputs are concatenated and fed into a neural network to predict the probability of an output word over the vocabulary of words. In the training process, to predict the nth word in the generated captions, the words from the target caption upto (n-1)th words are made available to the LSTM encoder. Therefore, training samples contain, extracted image features, its caption till (n-1)th position,and the nth word as target. For example, an Image with ID abcd has the target caption "There are trees in the garden.". This will be converted to:

ID	Input Caption	Target
abcd	'startseq'	there
abcd	'startseq' there	are
abcd	'startseq' there are	trees
abcd	'startseq' there are trees	in
abcd	'startseq' there are trees in	the
abcd	'startseq' there are trees in the	garden
abcd	'startseq' there are trees in the garden	'endseq'

While generating captions, an image and a 'startseq' is fed into the model to start the word generation process. Each generated word is then appended to the previously generated words for the image until either the 'endseq' is generated or the maximum length of the caption is reached.

3.2. Improvements in Model

To improve the above model, I tried to implement the Visual Attention model [23] proposed by Kevin Xu et al. in "Show, Attend, and Tell". In this model, the image features will be extracted from the output of the 4th convolutional layer block of VGG19, and then fed into CNN layer, and then finally to an LSTM layer with Visual Attention. The CNN layer is included to ensure that can be transformed for this task. All the preprocessing steps, as done for the previous model, were done for this model as well, i.e. a tokenizer was learnt on the captions and the captions were converted to a sequence of numbers. After obtaining the feature vectors for each image in the training dataset, the feature vectors and captions were combined to form the new training dataset. This model uses an attention block, which computes the similarity score between the hidden state of the LSTM and the output of the CNN layer, which is then used to evaluate the context vector that is fed to the LSTM. Therefore, for generating each word, the LSTM has information about a particular region via the context vector, the previous hidden state, and the current input. This method helps in learning Visual Attention i.e. which parts of the image to concentrate on while generating the captions. The model has been show in Figure 4.

The equations of the LSTM will now be,

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t, \hat{z}_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t, \hat{z}_t] + b_i)$$

$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t, \hat{z}_t] + b_c)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t, \hat{z}_t] + b_o)$$

$$h_t = o_t \cdot tanh(C_t)$$

where, only $\hat{z}$, the context vector is new, rest of the symbols are the same as previously mentioned in Section 2.2.

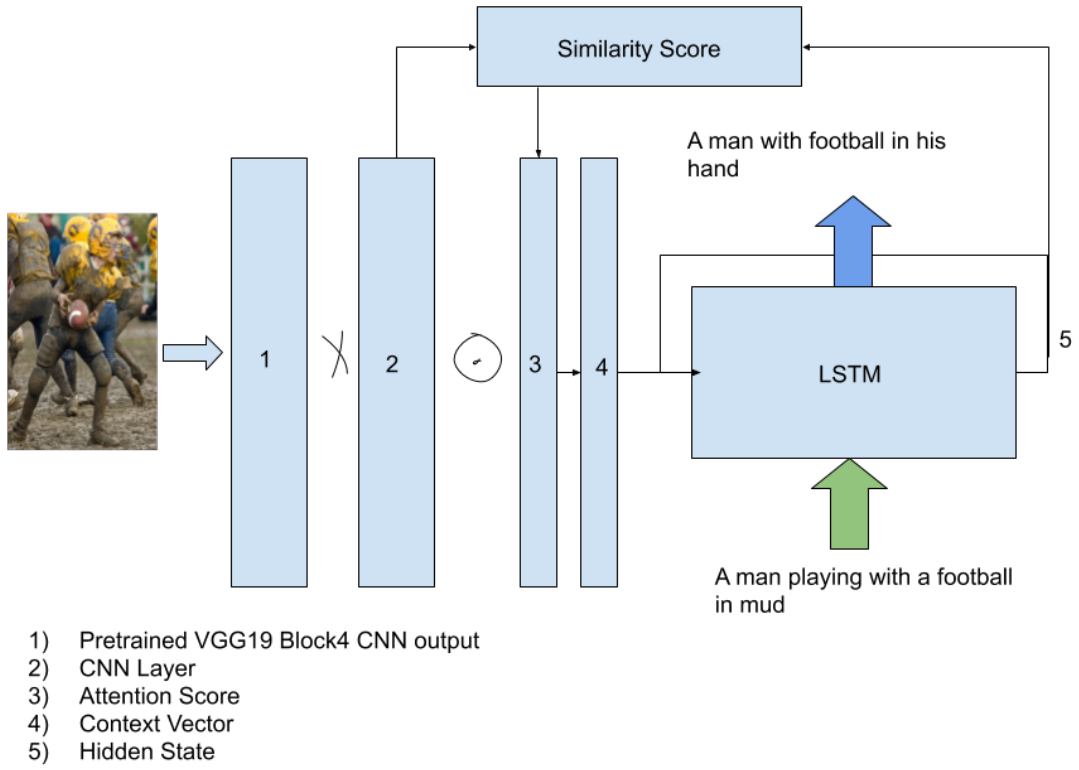

Figure 4. A Deep Learning Model using Visual Attention for Caption Generation.

Describe how attention scores are calculated, and then how the context vector is calculated. The similarity scores is calculated using the hidden state of the LSTM and the CNN output. For this, the two vectors are concatenated and transformed as below,

$$e_{ti} = score(h_t, h_i) = v_a^T tanh(W_a[h_t; h_i])$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L} \exp(e_{tk})}$$

Where, α_{ti} are the normalized attention scores, h_t is the hidden state of the LSTM, and h_i is the representation of a part of the image. h_i can be obtained from the feature vectors extracted from the VGG19 model. Once, the attention weights are calculated as above, the context vector can be obtained by using the Soft Attention model proposed by Bahdanau et al.[2].

$$\hat{z_t} = \sum_{i=1}^{L} \alpha_{ti} a_i$$

This context vector is then fed into the LSTM for generating the next word. The training and caption generation for this model are similar to the previous model.

4. Datasets and Evaluation Metrics

4.1. Datasets

There are three main Datasets available for Image Captioning, MS COCO[13] Dataset and the Flickr30K[17], and the Flikcr8K[8] Dataset. The Microsoft MS COCO Dataset has 300,000 images with 5 captions per image, and 80 object categories. The Flickr30K Dataset contains 30K images collected from Flickr, with 158K captions, where as, Flickr8K Dataset contains 8K images with about 40K captions.

Due to limited computing resources, the models were trained on the smallest of these, the Flickr8K Dataset. It has diverse captions with conceptual descriptions of the entities in the image. It has 6K images for training, with 1K images for validation, and 1K images for testing.

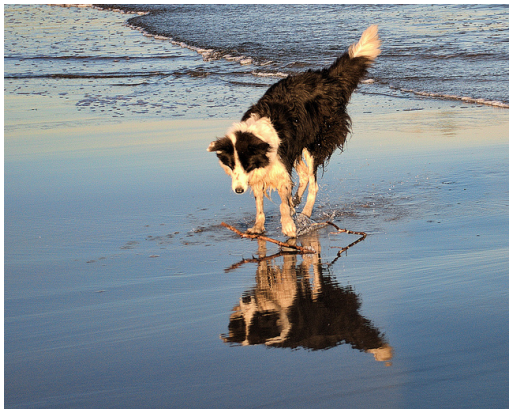

Figure 5. Generated Caption by Preliminary Model: startseq dog is running through the water endseq.

4.2. Evaluation Metrics

BLEU (Bilingual Evaluation Understudy)[16] is an evaluation metric for measuring the quality of machine generated text. It is based on the comparison of generated text with a set of reference texts. In evaluating the score, syntactical correctness is not considered. It is popular because it was a pioneer in automated evaluation of machine generated/translated text and also because it has reasonable correlation with human judgements of quality.

BLEU-n score is calculated for different weights given to 1-gram, 2-gram, 3-gram, and 4-gram word pairs generated in comaprison to the reference text. BLEU-n score weights are 1/n distributed i.e. for BLEU-1 it will be (1,0,0,0), BLEU-2 (0.5,0.5,0,0), BLEU-3 (0.33,0.33,0.33,0), and BLEU-4 will be (0.25,0.25,0.25,0.25). A BELU score as close to 1 is considered to be better, although it is not possible to get perfect scores even for Human Annotators.

5. Results

For the preliminary approach, the BLEU scores after generation of captions for 1K test images are:

BLEU-n	**Score**
BLEU-1:	0.544823
BLEU-2:	0.296060
BLEU-3:	0.195846
BLEU-4:	0.083903

An example caption is generated for an image, not present in the test set, is shown in the Figure 5.

The Attention Model made for the project had some dimension mismatch and as a result of which, it only predicted 'endseq' as the most probable word at testing time. It did learn, as can be shown by the training loss in Figure 6. As a result of the error, the attention mapping on the images is also incorrect. An example caption generated

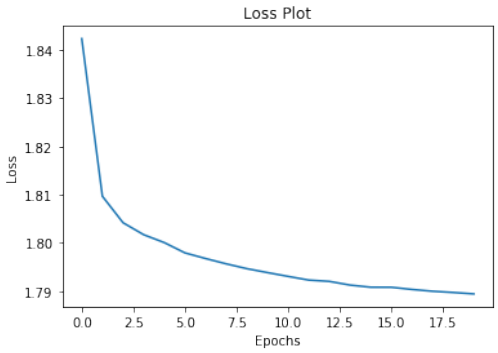

Figure 6. Training loss vs Epoch plot for the Attention Model. It decreases with every epoch.

Figure 7. An image from the validation set. The captions generated by the attention model are shown in Figure 8.

for an image from the validation set is shown in Figure 7 and Figure 8. The generation process stops at the 'endseq', but I bypassed it to generate the full caption length to see if it generated any other words. It didn't. Another example image from the validation set is shown in Figure 9.

6. Future Work

I plan on fixing the dimension mismatch error in the attention model in the coming days. Apart from that, this model can further be improved by the use of a pre-trained Language Model, which will be fine-tuned for this task. Also, the use of Transformers[19] to boost training speeds of attention models has been reported.

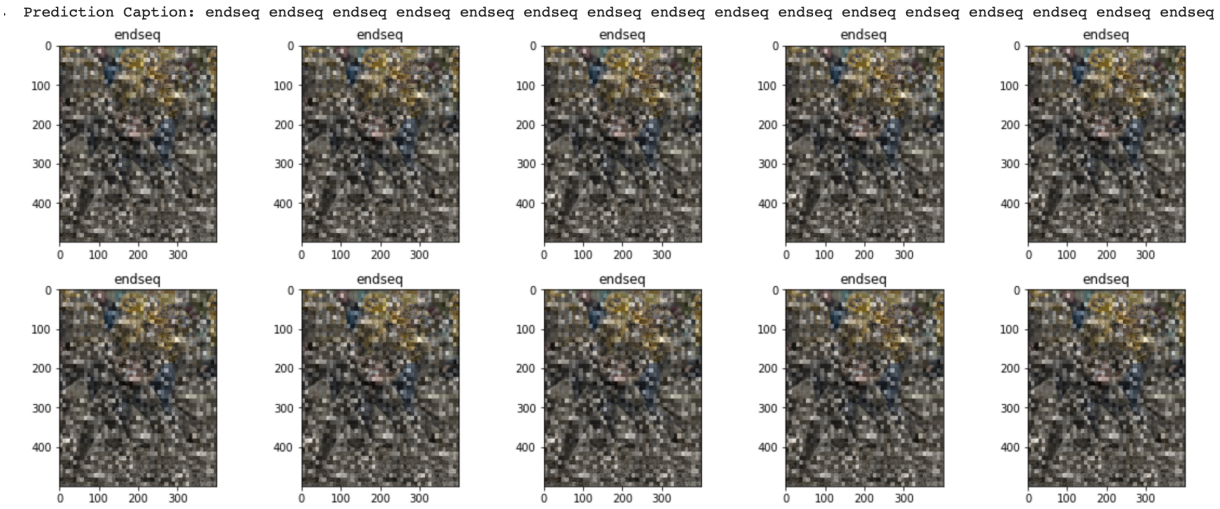

Figure 8. Captions generated for Image shown in Figure 7.

Figure 9. An image from the validation set along with the captions generated by the attention model.

7. Code Repositories Referred

I referred the Blog post by Jason Brownlee[3] to make the preliminary model and to learn about Text Generation. I referred the TensorFlow Tutorial [1] on Image Captioning with Visual Attention to make the Attention model. I also referred to the github repositories [5] and [22] to implement the data requirements for the Attention Model.

References

[1] Image captioning with visual attention. Accessed on: 2019-11-30. 7

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 5

[3] J. Brownlee. How to develop a deep learning photo caption generator from scratch, Oct 2019. 2, 7

[4] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5659–5667, 2017. 2

[5] Y. Choi. Choi's show-attend-and-tell. Accessed on: 2019-11-28. 7

[6] B. Dai, S. Fidler, R. Urtasun, and D. Lin. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2970–2979, 2017. 2

[7] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1

[8] M. Hodosh, P. Young, and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *Journal of Artificial Intelligence Research*, 47:853–899, 2013. 5

[9] M. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 51(6):118, 2019. 1, 2

[10] J. Jin, K. Fu, R. Cui, F. Sha, and C. Zhang. Aligning where to see and what to tell: image caption with region-based attention and scene factorization. *arXiv preprint arXiv:1506.06272*, 2015. 2

[11] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4565–4574, 2016. 2

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In

Advances in neural information processing systems, pages 1097–1105, 2012. 1

[13] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[14] J. Lu, C. Xiong, D. Parikh, and R. Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 375–383, 2017. 2

[15] C. Olah. Understanding lstm networks, Aug 2015. 4

[16] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002. 6

[17] B. A. Plummer, L. Wang, C. M. Cervantes, J. C. Caicedo, J. Hockenmaier, and S. Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015. 5

[18] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 3

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 6

[20] C. Wang, H. Yang, C. Bartz, and C. Meinel. Image captioning with deep bidirectional lstms. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 988–997. ACM, 2016. 2

[21] Z. Wu and R. Cohen. Encode, review, and decode: Reviewer module for caption generation. *arXiv preprint arXiv:1605.07912*, 2016. 2

[22] K. Xu. arctic-captions. Accessed on: 2019-11-28. 7

[23] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015. 2, 4