Three Models: LSTM, Attention based LSTM and Transformer for Image Caption

Abstract

Inspired by recent work in image caption, attention and translation, our paper proposes three models to deal with image caption problems. The first model uses ResNet 152 as the encoder and LSTM as the decoder. Based on the first model, the second model uses the attentitive LSTM as the decoder. The third model is inspired by Transformer Model proposed by Google. Our paper describes the three models in detail and represents algorithms, architectures of our models. Codes are provided in appendix (surely original). We use Flickr30K as the dataset and metric BLEU, METEOR and CIDEr are used to evaluated our models. Finally, our models are compared with others' models in Part Analysis and some expectations about future work are also made in Part Analysis.

1 Introduction

People could depict contents of pictures easily and accurately, concentrating on some objects and ignoring the others. However, generating a sentence level description for an image is not an easy task for visual recognition models. The model not only needs to distinguish different objects, but also needs to combine related words and form a subtle, accurate sentence.

In recent years, various fanscinating methods have been proposed to generate as accurate and human-like as possible captions of an image automatically. Fortunately, the technology of obtaining sentence level descriptions has developed so much since attention system was applied in image caption model, which renders sentences generated by the model pretty satisfying and the brief development procedure is shown in Part Two(Related work).

Visual recognition models have various applications, such as navigation for the blind, image retrieval and early childhood education and so on[14], which releases human-beings from tough work.

Our paper has three models. Model One uses a Residual Network 152 as the encoder and a LSTM[9]; Model Two uses a Residual Network 152 as the encoder and an attention based LSTM as the decoder; Model Three uses the transformer model proposed by Google [17].

2 Related work

Previous models are provided in this section.

In recent 20 years, various models have been proposed. The idea of tranlating pictures into sentences was first proposed by Mori [15]; the triplet model [7] builds three spaces: image space, sentence space and meaning space (meaning space is formed by tripets <objection,action,scene>) and trains two maps (from image space to meaning space and from sentence space to meaning space), using method Markov Random Field (MRF); Girish et al.[11] constructed a Conditional Random Field (CRF) to detect candidate objects, to process candidate region by a set of attribute classifiers and to generate predicted labels.

Models proposed above are really complex and thanks to the rapid development of neural networks, generative models were simplified to a great extent. Some articles built models [10, 2] that use neural networks (such as CNN,RNN,m-RNN and so on) to embed images and words into a multimodal space. Then embedding structure was simplified to an end-to-end structure[4, 14], which directly models the probability distribution of generating a word given previous words and an image. Google article[20] proposed NIC model, based on a neural network consisting of a vision CNN followed by a language generating RNN, which has a tremendous impact in this field. Besides, there are also many models using Reinforcement Learnig, such as SCST [16].

However, sentences produced by the above models were not human-like enough and could not concentrate on the major objects and events in an image. Then Kelvin et al.'s attention based model [22] solved these problems better, on which our Model Two is mostly based. There are also many previous models incoporating attention into neural networks to deal with vision related tasks[21, 6, 12] and so on. Specifically, Jiasen Lu et al's work [13] updates Girish Kulkarni et al' work [11] mainly by adding attention system to the system; Long Chen et al's work [3] proposes the spacial and channel-wise attention in convolution networks; Peter Anderson et al's work [1] combines bottom-up and top-down attention with detection method (Faster-RCNN), a thoroughly state-of-the-art, which achieves the highest CIDEr score 120 so far.

Even though Kelvin et al.'s attention based model [22] performs well in generating image captions, a major problem still remains in plain CNN network. That is increasing network layers could lead to larger error. Fortuantely, Kaiming He et al.'s work [8] could solve this problem. Thus, Model Two extracts both [8, 22] papers' essences, using Residual Network as the encoder and adding attention system to LSTM the decoder and Model Three uses Google's transformer model [17] (with no encoder).

3 Model

Our paper has three models. Model One and Model Two both use ResNet152 as the encoder and use Long Short-Term Memory Network (LSTM), attention based LSTM as the decoder respectively. Model Three uses the transformer

model proposed by Google[17] (no encoder).

In Model One and Model Two, images are encoded in a vector with Residual Network 152 (a residual network with 152 layers) and captions are decoded from the vector produced by its corresponding image using a Long Short-Term Memory Network (LSTM) or an attention based LSTM. A raw image is input into our models and then a representative vector $\mathbf{y}$ of a sentence is generated:

$$y = \{\mathbf{y}_1, ..., \mathbf{y}_L\}, \mathbf{y}_i \in \mathbb{R}^N$$

where L is the length of the sentence and N is the dimension of the vocabulary space.

3.1 Model One

3.1.1 Encoder One: Residual Network 152

We use a Residual Network 152 [19] (a Residual Network with 152 layers) to extract figure vectors denoted as annotation vectors from images:

$$x = \{\mathbf{x}_1, ..., \mathbf{x}_D\}, \mathbf{x}_i \in \mathbb{R}^M$$

where D is the number of extracted vectors from an image and M is the dimension of figure vectors. Besides, for each $i = 1, ..., D$, $\mathbf{x}_i$ represents part of the image.

3.1.2 Decoder One: LSTM

Gradient vanishing and exploding is a common problem of RNNs and here we use a particular type of recurrent nets, LSTM. Each word is generated based on hidden layer $\mathbf{h}_{t-1}$, previously produced words and the annotation vectors. The definition of gates and cell update and output are shown bellow: (For convinience, denote $\mathbf{x}_t$ as the input of the cell at time t)

$$\mathbf{i}_t = \sigma(W_{ix}\mathbf{x}_t + W_{ih}\mathbf{h}_{t-1})$$

$$\mathbf{f}_t = \sigma(W_{fx}\mathbf{x}_t + W_{fh}\mathbf{h}_{t-1})$$

$$\mathbf{o}_t = \sigma(W_{ox}\mathbf{x}_t + W_{oh}\mathbf{h}_{t-1})$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot tanh(W_{cx}\mathbf{x}_t + W_{ch}\mathbf{h}_{t-1})$$

$$\mathbf{h}_t = \mathbf{c}_t \odot \mathbf{o}_t$$

$$p_{t+1} = softmax(\mathbf{h}_t)$$

where matrices W are parameters to be trained and $\odot$ means dot product of vectors. And $\mathbf{i}_t$, $\mathbf{f}_t$, $\mathbf{o}_t$, $\mathbf{c}_t$ are input gate, foget gate, output gate and memory gate respectively. $\sigma(\cdot)$ represents sigmoid function.

Figure1 illustrates the structure of an LSTM cell. Because our LSTM model is the same as [22], for convienence, this figure is from [22].

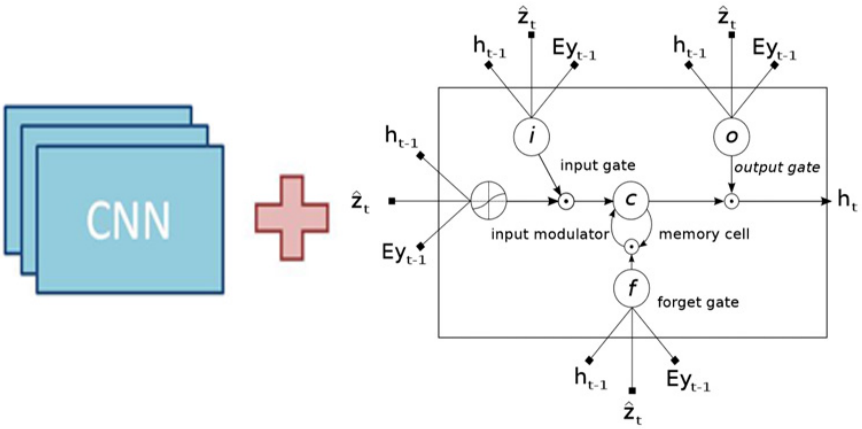

Figure 1: CNN and LSTM

3.2 Model Two

3.2.1 Encoder Two: ResNet 152

The same as Encoder One.

3.2.2 Decoder Two: Attention based Long Short-Term Memory Network

On the basis of Decoder One, we optimize the decoder by adding attention system:

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{pmatrix} T_{M+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \mathbf{z}_t \end{pmatrix} \tag{1}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{y}_t \odot \mathbf{g}_t \tag{2}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot tanh\left(\mathbf{c}_t\right) \tag{3}$$

where $\mathbf{E} \in \mathbb{R}^{m \times N}$ is an embedding matrix. $\mathbf{z}_t \in \mathbb{R}^M$ is the capturing vector, a dynamic representation of the relevant part of the image input at time t. A mechanism f_{att} is defined to compute $\mathbf{z}_t$ from annotation vectors $\mathbf{x}_i$(i=1,2,...,D).

For each i, α_i ($\alpha_i > 0$) means the probability that part i of the image is the correct part to concentrate for the next word and α_i is computed based on the previous hidden state $\mathbf{h}_{t-1}$ and $\mathbf{x}_i$, which accords with the fact that the next word depends on the former words. Fomula are as follows:

$$e_{it} = f_{att}\left(\mathbf{x}_i, \mathbf{h}_i\right) \alpha_{it} = \frac{exp\left(e_{it}\right)}{\sum_{j=1}^{D} exp((e_{jt}))} \tag{4}$$

where f_{att} is a multilayer perceptron whose parameters need to be trained. Then the context vector $\mathbf{z}_t$ is defined by soft attention mechanism[6]:

$$\mathbf{z}_t = \sum_{j=1}^{D} \mathbf{x}_j \alpha_{it}$$

3.3 Model Three

Transformer model was proposed by Google [17] in 2017. When we finish the first two models, we find that the architecture of encoder and decoder is essential for our model's performance. And ResNet152 is a state-of-the-art model in CNNs. However, LSTM is so trivial that it is far from generating human-like and satisfying sentences. Hence, we decide to build Model three.

Transformer proposed by Vaswani et al. [17] is based on attention mechanisms with no recurrence and convolutions at all. Papers and experiments show that the Transformer performs better in solving sequence problems with large or limited training data, which means transformer model could be a good decoder for our model. Figure2 illustrates its architecture.

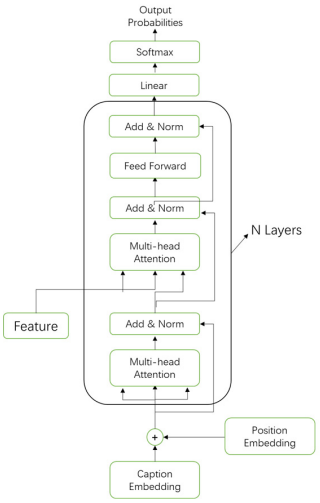

Figure 2: Transformer structure

3.3.1 Model Architecture

Different from using ResNet152 to extract new feature from scratch in the two above models, here in Model Three, we use the features provided by Prof. Fu. Since with Transformer, it takes much more time to train the model, we need to reduce the cost on extracting features. At each step, the decoder uses the image attention, positional attention and word attention to predict the next word in order to acquire a complete sentence. The Transformer Model consists of a stack of N identical layers and each layer has three sub-layers. The first layer uses the multihead mechanism and the input of the layer is the caption embedding. The second layer is a multi-head attention layer, which aims to correlate the text information and the image features with the attention mechanism. The third layer is a position-wise fully connected feed-forward network. Besides, there is a residual connection around each of the three sub-layers, combined with layer normalization. At the top, to get the probabilities for the sentence, we construct a fully connected layer and a softmax layer to convert the output vectors from the first three layer to probabilities. What's more, another difference between LSTM and Transformer is that all the words in the sentence can be parallelly generated by Transformer but LSTM can't.

3.3.2 Text Embedding

Word embeddings is a kind of word representation that allows words with similar meanings to have similar representations. Firstly, the frequency of words in the caption sentence will be calculated. Secondly, based on the frequency, each word will be represented by index. Thirdly, '<start>', '<end>' and '<UNK>' are the additional words. With these words represented by index (onebyone), we can get a dictionary, by which we can get index from words and know the word from its index. Then, let the one-hot vector x represent for each word when we train our model. With embedding model the one-hot vector will be embedded into a d model-dimensional vector. For an input sentence, it will be transferred to a (L, d) matrix as input for the model, where L means the length of the sequence and d denotes the dimension of vector .

3.3.3 Attention in transformer

Attention structure is shown in Figure3.

•Dot-product Attention

The Transformer uses the Scaled Dot-Product Attention. The input consists of three inputs, which are keys K, values V and queries Q. The attention is shown as follows:

$$Attention\,(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where $Q \in R^{L\times d_k}$ represents the query, $K \in R_{l\times d_k}$ means the key, $V \in R^{L\times d_v}$is the value and the L is the length of the sequence. Besides, $QK^T \in R^{L\times L}$ is the product operation, which could make the result extremely large or small

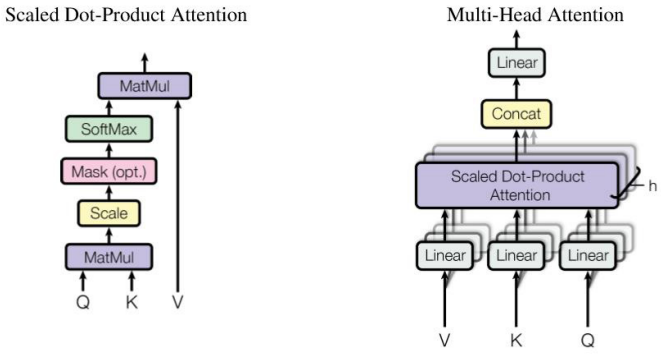

Figure 3: Attention structure

and hence will effect the precision of the variable. Thus $\sqrt{d_k}$ is used to scale QK^T. Finally, we can compute $Attention\,(Q, K, V) \in R^{L\times d_v}$ by the formula. Dot-product attention has obvious advantages: efficient in speed and space; can be computed by the optimized matrix multiplication method.

K and V is the key-value pair. At the first sub-layer of the Transformer, this pair is the text embedding matrix. At the second sub-layer, this pair is the image embedding matrix.

•Multi-Head Attention

For better performing attentions, the multi-head attention is composed of n scaled dot-product attentions. The multi-head attention is shown as follows:

$$h_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right)$$

$$H = Concat\,(h_1, ..., h_n)$$

$$O = HW_h$$

where matrices $W_i^Q \in R^{d_{model}\times d_k}$, $W_i^K \in R^{d_{model}\times d_k}$, $W_i^V \in R^{d_{model}\times d_v}$. $Q \in R^{L\times d_{model}}$,$Q \in R^{L\times d_{model}}$, $V \in R^{L\times d_{model}}$ are the inputs of the attention model. $h_i \in R^{L\times d_v}$ is the scaled dot-product attention's output. Besides, *Concat* is the concat function and *Attention* is the scaled dot-product attention. n scaled dot-product attentions are concentrated to generate $H \in R^{L\times(n\times d_v)}$. And $W_h \in R^{(n\times d_v)\times d_{model}}$ is used to project H into the output $O \in R^{L\times d_{model}}$.

At the first muliti-attention layer, keys, values and queries are the same matrices, which called the Self-Attention. At the second decoder layer, the keys and the values are the matrices from image feature, which store the image information. The queries are the outputs by the first decoder layer, which reserve the information in sentences.

4 Training

4.1 Model One and Model Two

As we all know, LSTM predict the possibilities of the next word according to image information, current state and predicted word, which is defined by $p\left(S_t|I, S_0, ..., S_{t-1}\right)$, where I is image feature and S_i means the word generated at time i (and S_i could be the ground truth word or the word predicted by our models). We use image feature and additional string '<start>' as the input for our decoder to initialize our the origin state before we feed our decoder with sentence caption. And different from the origin paper Show and Tell[22], we use a hyperparameter called 'use_tf' to control whether we feed the LSTM with ground true word or the one predicted by our model as current_word to predict the next word. Authors in the original paper only use ground truth words to get the softmax scores for the sentence predicted. However, we think that this method can reduce the robustness of the model, because the trained model would depend much more on the predicted word, which means it could lead to the completely wrong sentence if the predicted words are not accurate. So we set the 'use_tf' and compare it with a random number to determine whether to use the ground truth word or the predicted word at time t to predict the next word at time t+1.

4.2 Model Three: Transformer

Same with the original transformer in machine translation, our Transformer model can be trained in parallel. With image features matrix and the ground-truth sentence embedding matrix as input, the model get the probability distribution p(S'|S,I) where I denotes the image feature, S is the ground-truth sentence embedding matrix and S' is the sentence generated by the model. Now we gets the whole sentence probability for the current image which is same as the first two model.

4.3 Loss

The model would be trained by minimizing the cross-entropy loss which is the same as the one described in the NIC model, which is the sum of the negative log likelihood of the correct word at each step as follows:

$$L\left(I, S\right) = -\sum_{t=1}^{N} logp_t\left(S_t\right)$$

5 Inference

During the inference process of all models, the words will be generated one at a time. We can get the possibilities of the predicted word by a $1 \times len(vacab)$ vector. And we can use the greedy method or the beam search method to select

the possible word. As for greedy method, we choose the word with maximum possibility. As for BeamSearch method, every step, we can sort all candidate sequences in ascending order by their scores and select the first k as the most likely candidate sequence as the output at time t, which will be used to predict the next sequence at time t+1. Figure4 is randomly selected from our results.

Figure 4: A selection of sample results from our model

6 Experiments

We need to generate image caption sentences of test datasets and we load them into a json file. With json file and ground-true caption sentences file in pkl format, we use the tylin/coco code to evaluate the results.

In our model, the dimension of word embedding vectors is set to be 256 and the dimension of all lstm hidden states is set to be 512. In the Transformer model, the number of layers is set to be 6. Besides, the mini-batch is set to be 64 with the Adam method. The initial learning rate of the Transformer is 1e-3. During inference, we set the beam size to be 2 in the Transformer model.

6.1 Evaluation

Our paper uses BLEU(B1,B2,B3,B4), METEOR and CIDEr metric[18].

6.2 Results

Flickr30k contains 31,783 images, most of which depict humans performing various activities. And every image is paired with 5 captions. We trained our model with trained dataset and validation dataset. Then we use the trained model on the test dataset to get results. Results are shown in Table1.

Table 1: Results

Method	Flickr30k					
	BLEU-1	**BLEU-2**	**BLEU-3**	**BLEU-4**	**METEOR**	**CIDEr**
Deep VS[10]	0.573	0.369	0.240	0.157	0.153	0.247
Hard-Attention[22]	0.669	0.439	0.296	0.199	0.185	-
$ATT - FCN^{\dagger}$[23]	0.647	0.460	0.324	0.230	0.189	-
Spacial[12]	0.644	0.462	0.327	0.231	0.202	0.493
Adaptive[12]	0.677	0.494	0.354	0.251	0.204	0.531
Ours-Simple LSTM	0.587	0.351	0.212	0.158	0.153	0.176
Ours-LSTM with attention	0.614	0.402	0.267	0.193	0.184	0.226
Ours-attention with Beam width 2	0.624	0.415	0.253	0.196	0.189	0.244
Ours-Transformer-inspired	0.643	0.420	0.285	0.203	0.193	0.305

7 Conclusion

Compared with the origin paper, we don't use many tricks and do enough parameters adjusting work, for which our model does not perform well. To be honest, our models perform no better than the state-of-the-art models.

8 Analysis

Our models don't get rid of the architecture of Encoder-Decoder. And we do some modification to make our first two model different from the original model proposed in other papers, such as using predicted word in our LSTM instead of ground-truth words. And we get the scores closed to the results in the papers. What's more, we built the third model to improve the performance. Same as some related papers, we are trying to use better encoder or decoder to improve our models' performance, which means using transformer in decoder. And based on the results above, it does contribute to improvement of Cider scores. So it means we can improve the Cider score by updating our decoder or encoder continually. As we all know, the bert model[5] has achieved great success in various NLP tasks, which is a fantastic model. So we think it will be interesting to use it in our model. Besides, we can see the image caption task as the generator, so reinforcement learning method will have a good future in this field. Recent papers only use reinforcement learning method to optimize

the training process instead of replacing the main model. With various Gan and VAE models proposed, we can try to use them in image caption task.

What's more, we also need to notice that transformer only performes much better than RNNs on long-sequence tasks, while generates several words generally in image caption tasks. Therefore, we can improve the models based on transformer and bert by optimizing their architectures to make them more special for image caption task. What's more, same as other papers, our model simply use the output from ResNet as image feature. However, the feature generated by ResNet can't convey full information of images, which is the point we can optimize. So I think there must be much future work for us to explore and the work can make sense.

References

[1] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and VQA. *CoRR*, abs/1707.07998, 2017.

[2] Karpathy Andrej, A. Joulin, , and L. Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. *International Conference on Neural Information Processing Systems MIT Press*, 2014.

[3] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, and Tat-Seng Chua. SCA-CNN: spatial and channel-wise attention in convolutional networks for image captioning. *CoRR*, abs/1611.05594, 2016.

[4] Xinlei Chen and C.Lawrence.Zitnick. Learning a recurrent visual representation for image caption generation. *CVPR*, 2015.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[6] Bahdanau Dzmitry, Cho Kyunghyun, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv:1409.0473*, 2014.

[7] Ali Farhadi and Mohsen Hejrati. Every picture tells a story: Generating sentences from images. *Computer VisionECCV 201match0. Spmatchringer Berlin Heidelberg*, 2010.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[9] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[10] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. *IEEE*, 2015.

[11] Girish Kulkarni and Visruth Premraj. Baby talk: Understanding and generating image descriptions. *IEEE*, 2012.

[12] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via A visual sentinel for image captioning. *CoRR*, abs/1612.01887, 2016.

[13] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Neural baby talk. *CoRR*, abs/1803.09845, 2018.

[14] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Zhiheng Huang. Deep captioning with multimodal recurrent neural networks (m-rnn). *ICLR*, 2015.

[15] Y. Mori, H. Takahashi, and R. Oka. Image-to-word transformation based on dividing and vector quantizing images with words. *WMISR*, 1999.

[16] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. *CoRR*, abs/1612.00563, 2016.

[17] Ashish Vaswani, Noam Ahazeer, and Niki Parmar et al. Attention is all you need. *NIPS*, 2017.

[18] Ramakrishna Vedantam, C.Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. *IEEE*, 2015.

[19] Andreas Veit, Michael Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. *NIPS*, 2016.

[20] Oriol Vinyals, Alexander, Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *CoRR*, 2015.

[21] Mnih Volodymyr, Hees Nicolas, Graves Alex, and Kavukcuoglu Koray. Recurrent models of visual attention. *NIPS*, 2014.

[22] Kelvin Xu, Jimmy Lei Ba, RyanKiros, Kyunghyun Cho, AaronCourville, Ruslan Salakhutdinov, RichardS.Zemel, and YoshuaBengio. Show, attend and tell: Neural image caption generation with visual attention. *Computer Science*, 2016.

[23] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 4651–4659, June 2016.