Project Final Report

Ye Tian
Stanford University
yetian1@stanford.edu

Tianlun Li
Stanford University
tianlunl@stanford.edu

Abstract

We plan to do image-to-sentence generation. This application bridges vision and natural language. If we can do well in this task, we can then utilize npl technologies understand the world in images. We plan to use datasets: Flickr8K, Flickr30K or MSCOCO. There are some existing works on this topic: [Karpathy and Fei-Fei], [Donahue et al.], [Vinyals et al.], [Xu et al.]. We plan to base our algorithm on that of [Karpathy and Fei-Fei] and [Xu et al.]. We plan also to evaluate our results with BLUE scores.

1. Introduction and Problem Statement

Automatically generating captions to an image shows the understanding of the image by computers, which is a fundamental task of intelligence. For a caption model it not only need to find which objects are contained in the image and also need to be able to expressing their relationships in a natural language such as English. Recently work also achieve the presence of attention, which can store and report the information and relationship between some most salient features and clusters in the image. In Xu's work, it describe approaches to caption generation that attempt to incorporate a form of attention with two variants: a "hard" attention mechanism and a "soft" attention mechanism. In his work, the comparation of the mechanism shows"soft" works better and we will implement "soft" mechanism in our project. If we have enough time we will also implement "hard" mechanism and compare the results.

In our project, we do image-to-sentence generation. This application bridges vision and natural language. If we can do well in this task, we can then utilize natural language processing technologies understand the world in images. In addition, we introduced attention mechanism, which is able to recognize what a word refers to in the image, and thus summarize the relationship between objects in the image. This will be a powerful tool to utilize the massive unformatted image data, which dominate the whole data in the world. As an example, for the picture on the right hand side, we can describe it as A man is trying to murder his cs231n partner with a clipper. Attention helps us to determine the relationship between the objects.

2. Related work

Work[3](Szegedy et al) proposed a deep convolutional neural network architecture codenamed Inception. The main hallmark of this architecture is the improved utilization of the computing resources inside the network. For example, our project tried to use layers "inception3b" and "inception4b" to get captions and attention. Because features learned from the lower layers can contain more accurate information of correlation between words in caption and specific location in image.

Work[4](Vinyals et al) presented a generative model based on a deep recurrent architecture that combined advances in computer vision and machine translation that can be used to generate natural sentences describing an image. The model is trained to maximize the likelihood of the target description sentence given the training image.Work[5](Jeff et al) introduced a model based on deep convolutional networks performed very good in image interpretation tasks. Their recurrent convolutional model and long-term RNN models are suitable for large-scale visual learning that is end-to-end trainable and demonstrate the value of these models on benchmark video recognition tasks.

Attention mechanism has a long history, especially in image recognition. Related work include work[6] and work[7](Larochelle et al). But until recently Attention wasn't included to recurrent neural network architecture. Work[8](Volodymyr et al) use reinforcement learning as a alternative way to predict the attention point. It sounds more like human attention. However reinforcement learning model cannot use back propagation so that not end-to-end trainable, thusly it is not widely use in NLP. In work[9] the authors use recurrent neural and attention mechanism to generate grammar tree. In work[10] the author use RNN model to read in text. Work[2](Andrej et al) presented a model that generates natural language descriptions of images and their regions. They combined Convolutional Neural Networks over sentences, bidirectional Recurrent Neural Networks over sentences and a structured objective that aligns the two modalities through a multimodal embedding. In Work[1](Xu, et al) attention mechanism is used in generation of image caption. They use convolutional neural network to encode image and use a recurrent neural network and attention mechanism to generate caption. By the visualization of the attention weights, we can explain which part the model is focusing on while generating the caption. This paper is also what our project based on.

3. Image Caption Generation with Attention Mechanism

3.1. extract features

The input of the model is a single raw image and the output is a caption $\mathbf{y}$ encoded as a sequence of 1-of-K encoded words.

$$y = \{\mathbf{y_1}, ..., \mathbf{y}_C\},\ \mathbf{y_i} \in \mathbb{R}^K$$

Where K is the size of the vocabulary and C is the length of the caption.

To extract a set feature vectors which we refer to as annotation vectors, we use a convolutional neural network.

$$a = \{\mathbf{a_1}, ..., \mathbf{a}_L\},\ \mathbf{a_i} \in \mathbb{R}^D$$

The extractor produces L vectors and each element corresponds to a part of the image as a D-dimensional representation.

In the work[1], the feature vectors was extract from the convolutional layer before the fully connected layer. We will try different layers such such as convolutional layers to compare the result and try to choose the best layers to produce feature vectors that contains most precise in formation about relationship between salient features and clusters in the image.

3.2. caption generator

The model use a long short-term memory (LSTM) network that produces a cation. At every time step , we will generate one word conditioned on a context vector, the previous hidden state and the previously generated words.

Using $T_{s,t} : \mathbb{R}^s \rightarrow \mathbb{R}^t$ to denote a simple affine transformation with parameters that are learned.(work[1])

$$\begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{Ey}_{t-1} \\ \mathbf{h}_{t-1} \\ \hat{\mathbf{z}}_t \end{pmatrix} \quad (1)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \quad (2)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot tanh(\mathbf{c}_t) \quad (3)$$

Where, respectively $\mathbf{i}_t, \mathbf{f}_t, \mathbf{c}_t, \mathbf{o}_t, \mathbf{h}_t$ are the input, forget, memory, output and hidden state of the LSTM. The vector $\hat{z} \in \mathbb{R}^D$ represents the context vector, capturing the visual information related to a particular input location. $E \in \mathbb{R}^{m \times K}$ is an embedding matrix. m and n is the embedding and LSTM dimensionality respectively. σ and $\odot$ are the logistic sigmoid activation and element-wise multiplication respectively.

The model define a mechanism ϕ that computes $\hat{z^t}$ from annotation vectors $\mathbf{a}_i, i = 1, ..., L$ corresponding to the features extracted at different image locations. And $\hat{z^t}$ is a representation of the relevant part of the image input at time $t4$. For each location i, the mechanism generates a positive weight α_i that can be interpreted as the relative importance to give to location i in blending the α_i's together. The model compute the weight α_i by *attention model* f_{att} for which the model use multilayer perceptron conditioned on the previous state h_{t-1}.

$$e_{ti} = f_{att}(\mathbf{a}_i, \mathbf{h}_{t-1}) \quad (4)$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\Sigma_{k=1}^{L} \exp e_{tk}} \quad (5)$$

After the weights are computed, the model then compute the context vextor $\hat{z^t}$ by

$$\hat{\mathbf{z}}_t = \phi(\mathbf{a}_i, \alpha_i), \quad (6)$$

where ϕ is a function that returns a single vector given the set of annotation vectors and their corresponding weights.

he initial memory state and hidden state of the LSTM are predicted by an average of the annotation vectors fed through two separate MLPs.

$$\mathbf{c}_o = f_{init,c}(\frac{1}{L}\Sigma_i^L \mathbf{a}_i)$$

$$\mathbf{h}_o = f_{init,h}(\frac{1}{L}\Sigma_i^L \mathbf{a}_i)$$

In the model, we will use a deep output layer to compute the output word probability given the LSTM state, the context vector and the previous word:

$$p(y_t | a, y_1^{t-1}) \propto \exp(L_o(Ey_{t-1} + L_h h_t + L_z \hat{z}_t)) \quad (7)$$

Where $L_o \in \mathbb{R}^{K \times m}$, $L_h \in \mathbb{R}^{m \times n}$, $L_z \in \mathbb{R}^{m \times D}$,and E are learned parameters initialized randomly.

3.3. Loss Function

We use a word-wise cross entropy as the basic loss function l_0. Further more, to encourage the attention function to produce more expressive output, we define l_1, l_2 as the variace of α_t along the sepence axis and spacial axise correspondingly. Then define the overall loss function as $l = l_0 + \lambda_1 l_1 + \lambda_2 l_2$, where λ_1 and λ_2 are hyperparameters.

4. Architecture

CNN features have the potential to describe the image. To leverage this potential to natural language, a usual method is to extract sequential information and convert them into language.. In most recent image captioning works, they extract feature map from top layers of CNN, pass them to some form of RNN and then use a softmax to get the score of the words at every step. Now our goal is, in addition to captioning, also recognize the objects in the image to which every word refers to. In other word, we want position information. Thus we need to extract feature from a lower level of CNN, encode them into a vector which is dominated by the feature vector corresponding to the object the word wants to describe, and pass them into RNN. Motivated by the work asdasdasdasd, we design the architectures below.

$\{a_i\}$ is the feature vector from CNN. We get these feature map from the inception-5b layer of google net, which means we have 6x6 feature vectors with 1024 dimensions. Firstly, we use function f_{init}, h and f_{init}, c to generate initial hidden state and cell state for the LSTMs. Input of LSTM0 is word embeddings. Input of LSTM1 is h0, which is the output of LSTM0, concatenated with attention vector z, which is an weighted average over $\{a_i\}$. The weight alpha is computed from the combination of h0, representing information of current word, and each of $\{a_i\}$, representing position information.

Our labels are only captions of the images. But to get a better caption, the network must force alpha to extract as much information as possible from $\{a_i\}$ at each step, which means alpha should put more weights on the area of the next word. This alpha is exactly the attention we want. Here for f_{init}, h, f_{init}, c, we use multilayer perceptrons(MLP). For f_{att}, we use a CNN with 1x1 filters.

To further reduce the influence of the image information as a whole and thus put more weight on attention information, we build a new model where we send $\{a_i\}$ directly to the first input of LSTM0 throug a MLP, and initialize h and c as 0. An even more extreme model is to only use z as information source from the image.

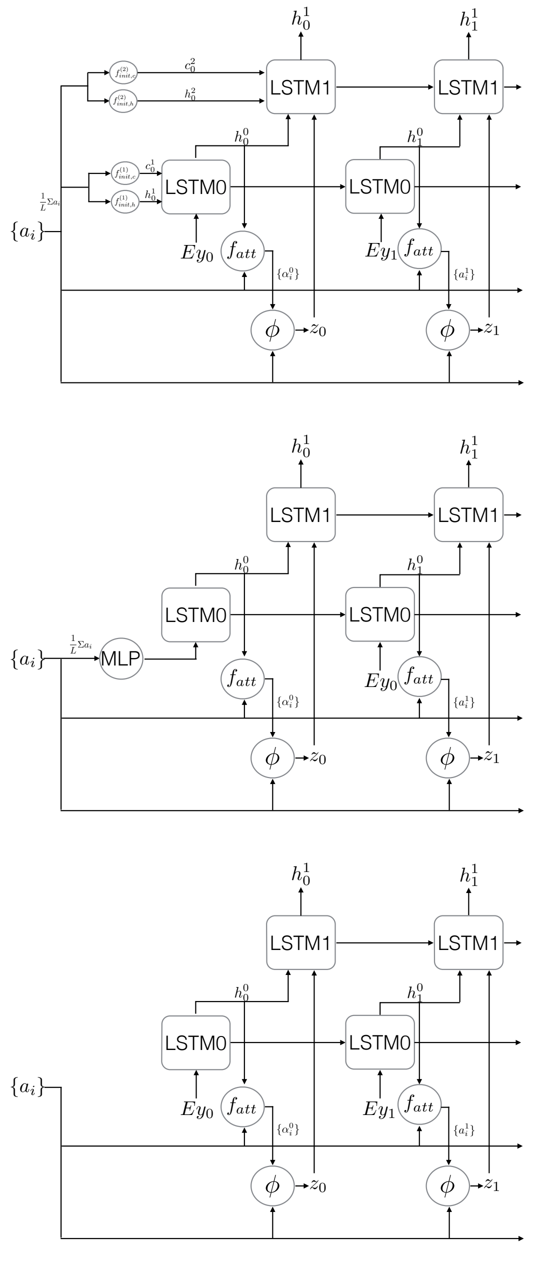
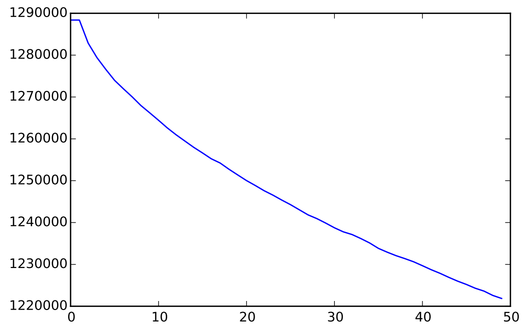


Figure 1: The first is original model. The second is Model with more weights on attention. The third is Model only depending on attention

5. Dataset and Features

We use dataset from MSCOCO. A picture is represented by a dictionary, the keys are as follow: [sentids, filepath, filename, imgid, split, sentences, cocoid]. Where the sentences contain five sentences related to the picture. We have 82783 training data and 40504 validation data to train and test out model.

For the sentences, we build a word-index mapping, add a “#START#” and “#END#” symbol to its both ends, and add “#NULL#” symbol to make them the same length. Because some words in the sentences are very sparse, when we generate a vocabulary from the sentences we need to set a threshold to decrease the classification error. The threshold should not only remove the spares words, but also avoid producing too many unknown words when predict the sentences. Thus, we observe the curve of vocabulary size – threshold and the total word size – threshold, which are showed in Fig 2. The previous one is exponential and



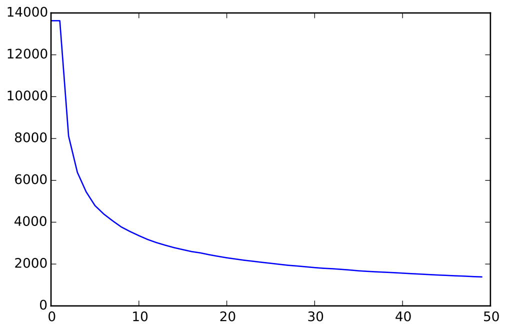


Figure 2: the upper figure is vocabulary size – threshold curve, the lower figure is total word count – threshold curve

the latter one is linear, so we choose 10 as our threshold. For images, we preprocessing them by cropping them to 224x224 and subtract the mean. Because they are already a large number of data, we dont do any data augmentations.

We tried to extract features of "inception3b", "inception4b" and "inception5b" from Googlenet. Among them, "inception5b" is relative higher than the previous 2 layers, thusly it contains less region information and hard to be trained to get attention. The layer "inception3b" is relative lower so that it contains more region information but less expressive to for caption.

6. Experiment, Result and Discussion

6.1. Experiment

In our architecture there are 2 parts. One is CNN encoder to map image to features, and the other is LSTM decoder with attention functions, which is a small CNN with 1x1 filters. We didnt finetune the encoder part and only trained the decoder part. To train the decoder, we used adam update. We tried learning rate from 1 to 1e-5 and found 5e-4 with decay rate 0.995 produce a best learning curve. Because feaure maps are smaller than images, and only decoder part was trained, so to make best of GPU memory, we used a large minibatch size of 512 samples.

At the beginning, we can overfit a small dataset with 1000 samples. But when we went to full dataset of 120,000 samples, we cannot overfit it even we increase the number of hidden units and depth of attention function. Then we adopted a gradually tuning method: train the model on dataset with size of 1000, 10000, 60000 and gradually pick our hyperparameters. Finally we got a good model with 60000 training samples, LSTM hidden size of 512 and MLP hidden size for [1024, 512, 512, 512], which generalize decently well.

6.2. Results

Some of our results are shown in Fig 4 and Fig 5.

We can see that the generated sentences expressed the pictures quite well. The main parts of the images can be recognized and shown in the sentence, and also of the minor parts are also encoded, such as the flower in the corner of xxxxxxxxx. Also, there are some mistakes such as the refrigerator and lamp in xxxxxxxxxx. But we human beings are also easy to make such mistakes since there do exist similar objects in the image. The generated sentences also do well in following grammar.

As for attention, our model is only able to recognize the most important part of the images. That is, the attentions at each step are the same. There are 2 major reasons. Firstly, since the features are input at the first step of LSTM, the overall information of the image has been feed into the decoder, which is enough to generate a decent sentence, and thus the following inputs can be coarser. This is exactly the motivation of our other models. They are potential to work better given more finetune. Secondly, the receptive field of inception 5 is quite large (139 x 139). So to focus on the main part of image is enough to get a good sentence. To address this problem, we can use lower level features from CNN with more expressive f_{att}, i.e., to deepen the f_{att} CNN and enlarge the number of hidden units in each layer.

We used BLEU score as quantitative metric for our results. We can see our model cannot achieve the state-of-the-art. To get better results, we should enlarge our model, train and tune it further.

Model	B-1	B-2	B-3	B-4
BRNN(work[2])	64.2	45.1	30.4	20.3
Google NIC	66.6	46.1	32.9	24.6
Log Bilinear	70.8	48.9	34.4	24.3
Soft-Attention(work[1])	70.7	49.2	34.4	24.3
Hard-Attention(work[1])	71.8	50.4	35.7	25.0
Our-model	44.4	30.3	20.0	14.8

Table 1: BLEU-1,2,3,4 metrics compared to other methods

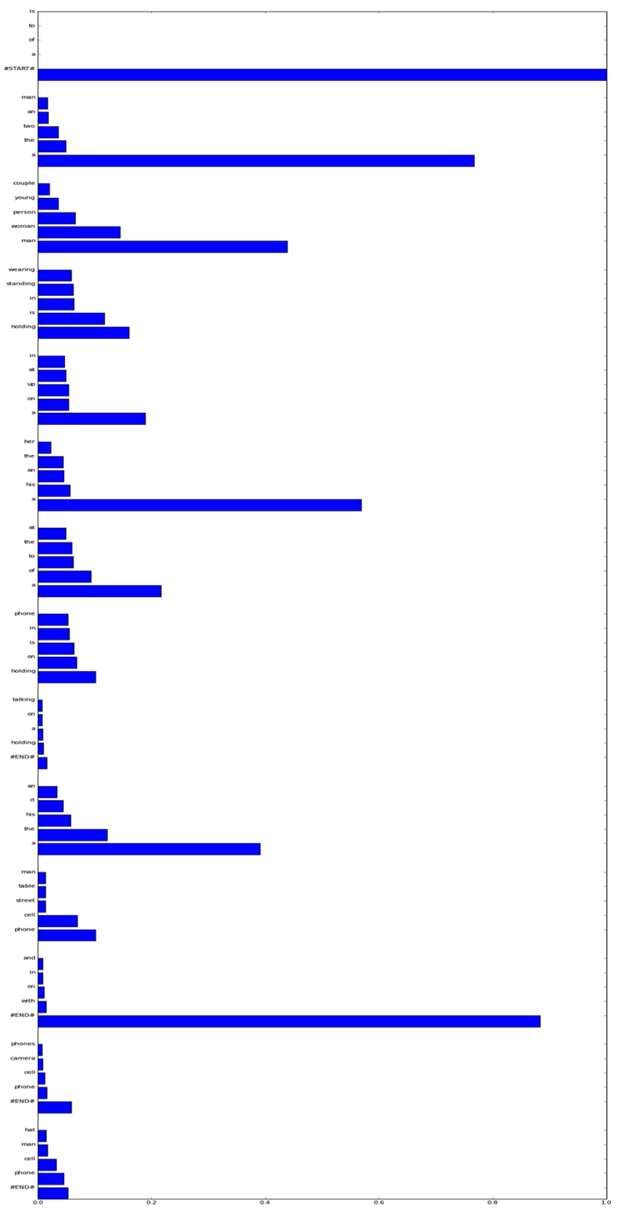


Figure 3: 5 most probable words generated by the model with one input image

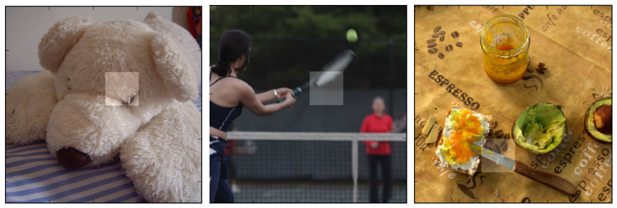
7. Furture Work

Advanced loss function: The original softmax loss function can cause problems. It can produce force negative. For example, if we input a the test picture with caption A man is riding a horse, the produced caption A horse is carrying a horse will produce high loss, but actually these two caption all correctly describe the picture. On the other hand the model can also produce force negative. For example, if the previous test picture produces a caption A man is carrying a horse, the loss will be small, but this is actually a wrong description of the picture.

Sharper attention: From the result we notice that the attention coefficient are evenly distributed, which means that the model takes the whole picture information to generate the next time step hidden layer via LSTM. But we expect that we can highlight specific part of the picture related to the certain word. To achieve this goal we can use hard attention, which restricts information extraction from image as whole. We can also use a harper activation function instead of softmax to produce a suitable attention distribution. Moreover, we can label more detailed captions to force the mode to attend smaller parts.

Language model: Since the model will produce a probability distribution of the vocabulary on every time step, we can use language model to generate natural sentences based on these vocabulary probability distributions. Further more, we can build a markov random field like hidden markov model on the top of the the softmax output layer.

We can try different architecture and especially layer of CNN encoder to get a better feature map level.

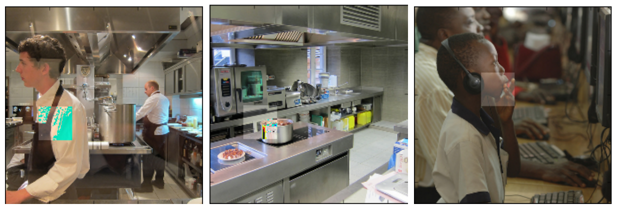


(a) a black bear with flowers resting on a bed

(b) a woman standing on top of a court with a tennis racquet

(c) a meal sitting in top of a table on a wooden table

Figure 4: Training Set



(a) a man are sitting at a kitchen area with a group inside

(b) a kitchen is filled with a lamp and refrigerator

(c) a woman standing on front of a phone and head to a house

Figure 5: Validation Set

References

[1] Xu, Kelvin, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio."Show, attend and tell: Neural image cap-

tion generation with visual attention." arXiv preprint arXiv:1502.03044(2015).

[2] Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions" In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3128-3137. 2015.

[3] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 1-9.

[4] Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan. "Show and tell: A neural image caption generator." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156-3164. 2015.

[5] Donahue, Jeffrey, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. "Long-term recurrent convolutional networks for visual recognition and description." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2625-2634. 2015.

[6] Larochelle H, Hinton G E. Learning to combine foveal glimpses with a third-order Boltzmann machine[C]//Advances in neural information processing systems. 2010: 1243-1251.

[7] Denil M, Bazzani L, Larochelle H, et al. Learning where to attend with deep architectures for image tracking[J]. Neural computation, 2012, 24(8): 2151-2184.

[8] Mnih V, Heess N, Graves A. Recurrent models of visual attention[C]//Advances in Neural Information Processing Systems. 2014: 2204-2212.

[9] Vinyals O, Kaiser , Koo T, et al. Grammar as a foreign language[C]//Advances in Neural Information Processing Systems. 2015: 2755-2763.

[10] Hermann K M, Kocisky T, Grefenstette E, et al. Teaching machines to read and comprehend[C]//Advances in Neural Information Processing Systems. 2015: 1684-1692.