IERG4999O Final Year Project II

Show, Attend and Tell:
Neural Image Caption Generation of Daily Objects

Chan Chi Hang
1155063984

A FINAL YEAR PROJECT REPORT
SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF BACHELOR OF INFORMATION ENGINEERING
DEPARTMENT OF INFORMATION ENGINEERING
THE CHINESE UNIVERSITY OF HONG KONG

May, 2018

Contents

Abstract
Neural image captioning has been one of primary research area of computer vision and machine translation in machine learning over the years. In this project, I have studied a new approach by using “attention” to take focus on the image and generating caption with our CNN and LSTM network, where the approach has achieved the state-of-art performance on the Flickr8k, Flickr30k and MSCOCO benchmark metrics recently.

1. Introduction

Different from simple object detection and recognition, automatically generating captions of an image also included a challenging task of scene understanding. Not only do we need to identify the main objects in the image but we also need to obtain their relationship so as the context of the event, which actually requires extra understanding of the spatial relation of the image. For example by object detection we may be able to tell there is a man, a clock, a platform and a track, but in image captioning, our goal is to summarize the event and to conclude that “a man is standing on a platform, waiting for a train”, where the train was never appeared in the image, we should also be able to interpret the event, which refers to a remarkable human ability to compress a huge amounts of salient visual information into descriptive language.

Most of the machine learning networks we have seen so far typically takes an input(image or sentences) and uses the whole input to predict an output(image or sentences most commonly) [1] , however in this situation, usually not the whole input is equally important to or contribute equally to the output, even worse, in most of the situation, there may be only a relatively small portion of the input(image) actually useful for the prediction(whether it is classification, or in our case the caption generation). In this case, not only the result may be less accurate but it also has a potential of including lots of redundant complexity.

In contrast, a recent improvement has tackle this problem by using a new technique called “attention” [2][3]. The core idea or ideal situation of “Attention” is to dynamically attend the salient features of a particular region that is helpful for the use of following prediction. Not only does it provides a great relieve of computational pressure that allows a more efficient and larger network feasible, it also improves the performance when we need to do prediction for a large image. There are also advantages of using attention for debugging and improving the model, that I will discussed in the experiment section. The original research suggested that the image caption generation is well suited to the encoder-decoder framework is because it is analogous to “translating” an image to a sentence [4].

2. Related Work

There has been varying approach towards the sentence generation problem, there are approaches that use n-gram-based language models [5], and later some improvement has been made by using maximum entropy language model instead [6], more recent image description works were using recurrent neural networks(RNNs) that built on those language model on certain level. There are also examples of approaches uses sentence templates, which they try to fitting words into predefined sentence template frames [7] which faces certain limitation due to the predefined structure. And later some linguistically expressive model has been proposed so resolved the problem [8].

There were also a lots of research interested in attacking the image caption generation problem, including the aided by advances in training neural networks [9] and large classification datasets [10] which their works has “significantly improved the quality of caption generation using a combination of convolutional neural networks(convnets) to obtain vectorial representation of images and recurrent neural networks to decode those representations into natural language sentences.” [4]

The similar approach to our study are the approaches that uses neural networks for caption generation, like the one by Kiros *et al*. proposed a multimodal log-bilinear model that was biased by the features from the image [11]. Mao *et al*. also uses a similar approach but replaced with a feed-forward neural language model [12]. Both Vinyals et al. [13] and Donahue et al. [14] use LSTM RNNs for their models, the latter one applied LSTM so as to allow their model generate descriptions for video.

The model I study that is able to attend the salient part of an image while generating its captions were inspired by the recent advances and success in caption generation in employing attention in machine translation [15] and object recognition [16][17], and is the direct extends of their works.

3. Methodology

The goal of my project is to learn, train and try to improve a show, attend and tell model that uses convolutional neural network(CNN) and long short-term memory(LSTM) network to accomplish caption generation for images of daily objects. That is, in effect, in each time it will takes in a single image and try to generate a caption(an English sentence) encoded as a sequence of 1-of-K encoded words which should be able to describe the main objects as well as the event happening(able to show a certain understanding of context) in the image, if it is not totally irrelevant to our training dataset.

$$y = \{y_1, \ldots, y_C\}, y_i \in \mathbb{R}^K$$

The y on the left side of the above equation that is a single vector represents our expected output, which is the sentence that describe the image, where K is the size of the vocabulary and C is the length of the caption, where in our case the size of K is 23110 and C is less than or equal to 15. Details of the dataset we used will be included in the experiment part of this report.

3.1 Image Caption Generation with Attention

3.1.1 Encoder: Convolutional Features

The model I studied for the project followed the research of Kelvin, *et al*. [4] which they uses a convolutional neural network(CNN) to extract a set of feature vectors(annotation vectors). The extractor produces a total number of L vectors where each of them is a D-dimensional representation of the different part of the image respectively.

$$a = \{a_1, \ldots, a_L\}, a_i \in \mathbb{R}^D$$

The features is then extracted from a lower convolutional layer, by only using a subset of feature vectors, it allows the decoder to selectively focus(weighed) on certain parts of the input image instead of the whole image.

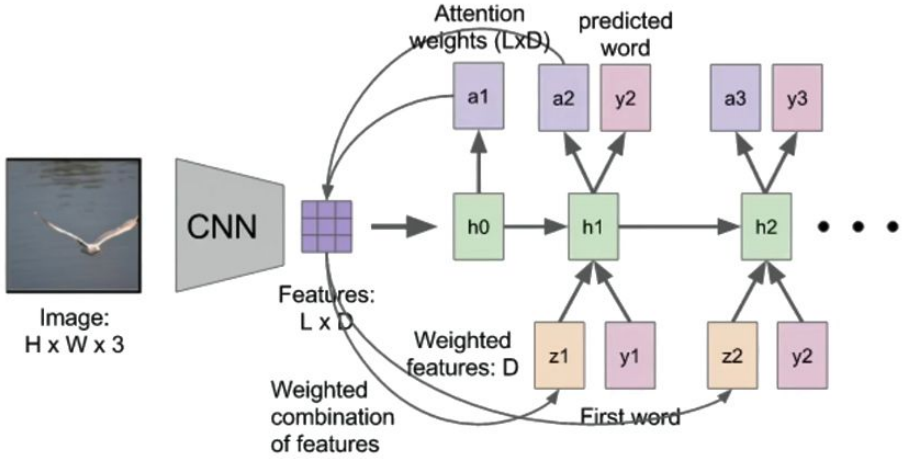


Figure 1. [18] An illustrative figure of the CNN and attention mechanism, instead of feeding the input image's features once to generate the words for output sentence, the attention approach finds a set of attention weights correspond to the feature vectors. Using the sum of attention weights will be also useful for predicting next and next word repeatedly of the sentence, and so as that the input is constantly changing throughout the process.

3.1.2 Decoder: Long Short-Term Memory(LSTM) Network

To generate a sensible and yet more accurate sentence that describes the image, a long short-term memory(LSTM) network [19] was being used, it produces a captions by generating one word at every time step conditioned on a context vector, the previous hidden state and the previously generated words. The implementation of LSTM is closely follows the one used in Zaremba *et al*. [20] (Fig. 1)

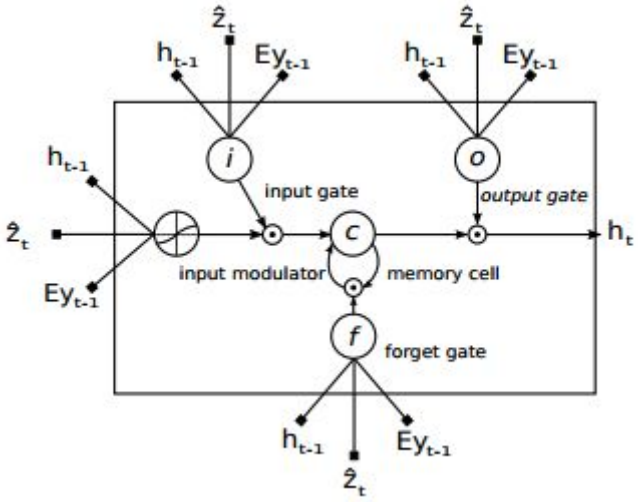


Figure 2. The mechanism of the LSTM cell. The square represent projections with learnt weight vector. Each cell learns how to weigh its input components and how to modulate that contribution to the memory. There is also a forget gate where it learns weights with erase the memory cell, and weights which control the output of memory. [4]

Where we can see the context vector $\hat{z}$ dynamically represents an input of different part of the image each time. For different location i, the mechanism generates a positive weight, which can be interpreted either as the probability that location i is the right place to focus for producing the next word (the “hard” but stochastic attention mechanism), or as the relative importance to give to location i in blending the positive weight together. Once the weight are computed(sum to one), the context vector will be computed based on the annotation vectors and their corresponding weights. Finally, there is a deep output layer [21] to compute the output word probability given the LSTM state, the context vector and the previous word.

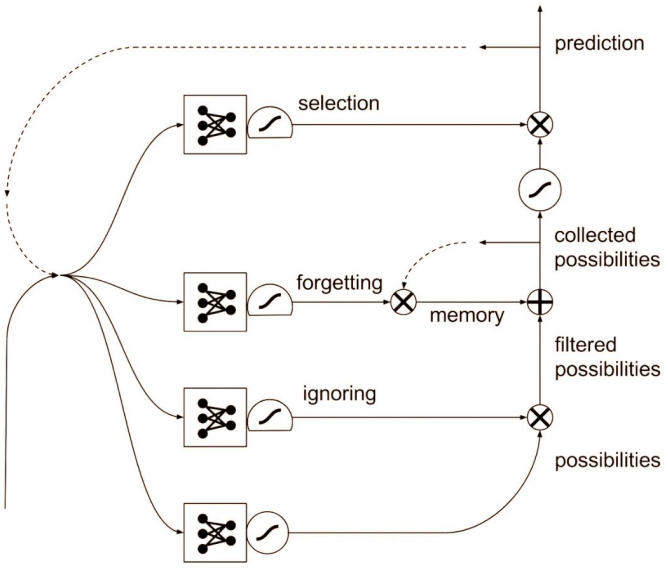


Figure 3. A illustrative figure of the LSTM combined with the model. The input was pass through the CNN layers for computation of features then the hyperbolic tangent (tanh) squashing function was applied, the LSTM also do a convolution and apply the logistic (sigmoid) squashing function, to decide which features are useful and place a weight for them, useful weight will then go through the plus junction with other factors while the ignored features will be eliminated by gating.

3.1.3 The Stochastic “Hard” and Deterministic “Soft” Attention

In fact, there are two ways of building the attention model. The easier one is called “soft” attention, where we predict the set of weight over the convolution layer, then try to select the marginal likelihood over the greatly weighted location and by summarizing all the locations using the weighted probability predicted, we will have a differentiable function that can be train with gradient descent(standard back propagation). Be aware that in this case all input are contributing the output, which actually kind of contradict to the fact that we want to selective focus on certain part of the image to reduce computation pressure.

The other is called “hard” attention, that it does a similar procedure but only take a sample of a subset of the input. By treating the attention locations as intermediate latent variables, a multinomial distribution can be assigned and an entropy term can be added to further reduce the estimator variance. It turns out that we will not have a differentiable function this time and the learning rule is equivalent to the reinforcement learning [22].

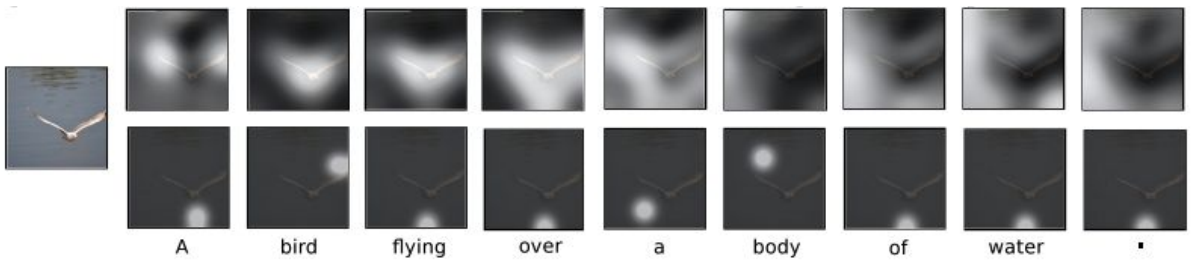


Figure 4. [4] Attention over time steps, each time a word was generated. The top row shows the effect of “soft” attention and the bottom row is the effect of “hard” attention, the output

sentence are exactly the same in this selected example, where the result could be different for a different image. The white part in the image represents the focus part(attention), which can be closely related to our visual intuition, that actually is a great advantage for debugging and improving the model, I will further discuss this in the experiment part.

3.2 Prepare and Training Procedure

The dataset used for training and validation is the MSCOCO data set(2014) [23]. In order to feed the image to the VGGNet19 Model, the images for training and validation is first resized into a fixed size of 224x224. During the preprocess, the caption vector was built from a index size of 23111 word dictionary, with a maximum length of 15. I used roughly 80000 images and 400000 captions for the training dataset and about 10 percent of it size for the validation and test dataset. Feature vectors(annotations) were then extracted from the VGGNet during this process.

Follow by the improvement mentioned by Kelvin, *et al*. [4], the performance was greatly improved, by building a dictionary mapping the length of a sentence to the corresponding subset of captions in preprocess, then randomly sample a length and retrieve a mini-batch of size 64 during the training. I have used the largest dataset available(MSCOCO) and the whole training process took roughly 1.5 days on two NVIDIA Titan Black GPU.

4. Experiments

4.1 The Image and Evaluation Dataset

The dataset used for training and validation is the comparatively(to Flickr8k and Flickr30k) large and challenging MSCOCO data set(2014) [23], which has a roughly number of 80000 images and 400000 captions for the training set.

4.2 Evaluation

As BLEU is the most frequently used metric, as being treated as the standard in caption generation literature, I also choose to use BLEU on the validation set to benchmark my model performance. The result from BLEU from 1 to 4 are as listed as follow. I also used another common metric METEOR [24] for a more reliable comparison.

Dataset	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
MSCOCO	CMU/MS Research [25]	-	-	-	-	20.41
	MS Research [26]	-	-	-	-	20.71
	BRNN [27]	64.2	45.1	30.4	20.3	-
	Google NIC [13]	66.6	46.1	32.9	24.6	-
	Log Bilinear [11]	70.8	48.9	34.4	24.3	20.03
	Soft Attention from paper [4]	70.7	49.2	34.4	24.3	23.9
	Hard Attention from paper [4]	71.8	50.4	35.7	25	23.04
	Our model	65.4	45.3	30.7	21.3	21.5

4.3 Quantitative Analysis

Due to the memory limit, I have almost halved the training features in my project, however we can still observe a great performance from the model. The original result from the paper we reference to has successfully achieved the state of art performance on the Flickr8k, Flickr30k and MSCOCO by that time, notice that are able achieve the performance using a single model [4].

Another point worth to mention is, by using the attention approach, we are able to extract a meaningful output image in the middle(as seen in figure 5). We can clearly observe or have a better guess at what causes the error, the advantage of able to visualizing the attention could be helpful to fix the mistake and overall improve the model.

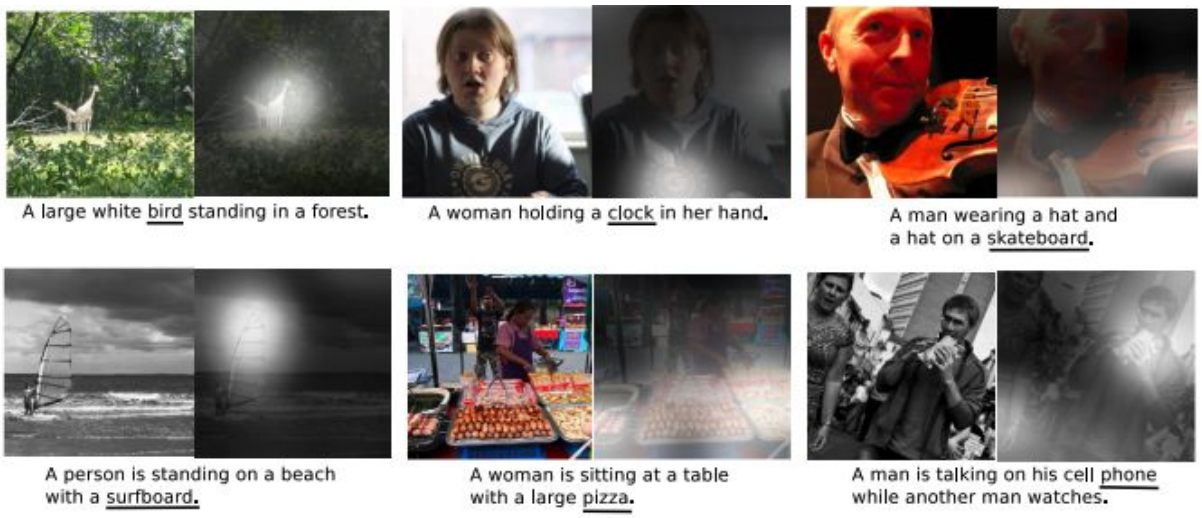


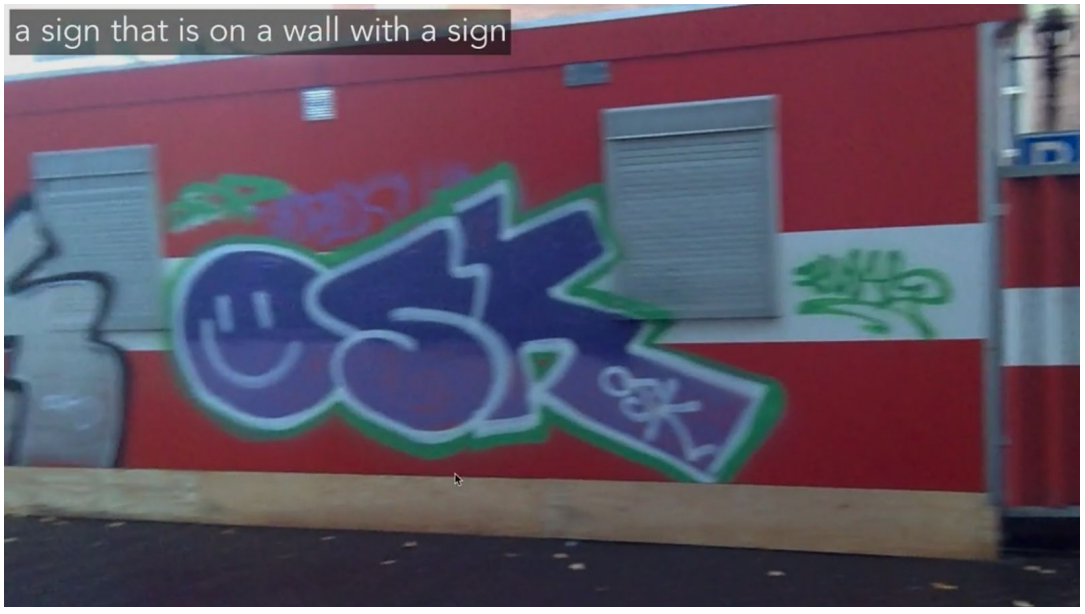
Figure 5. [4] Example of fail cases.

5. Application Research and Implementation

5.1 Captioning Application Today

Though the computer vision technology has many major improvements these years, there are still lacking of well spreaded real world application that utilize the technology that experts lately developed, specifically if when we talk about image captioning.

Some commonly known good application which utilized image captioning are probably the applications related to accessibility tools. The most stand out application are those reads out descriptive sentence for visually impaired people, like what NeuralTalk2 [28] did, their application in take live image feeds from mobile camera and generate descriptive sentence in real time, some other applications did implement to read out the sentence, the team I have mentioned even take out their product for testing for a whole day in Amsterdam, the result is satisfying, which give a visual sense of what normal people see every moment.



5.2 Building A Capioning Application

For my second semester research, I have taken the suggestions from last semester and decided to study and implement some interesting use cases and application for my researched model in the first semester, the first thing come to my mind is how to realize the model in a visible way, which is building application that really run on the trained model, so I can do further analyst to my model and also measure the actual performance in real world/different daily use cases.

5.2.1 Approaches

In order to implement my trained model into a useful application, basically what we need is to intake an input image, process with our trained model and return the generated sentence. It can be done in so many different ways, for example,

- we can build a http server, intakes input image as request, process and return sentence string response. The major advantage is we can easily building hooks to the application server across different platform, user can obtain the services via apis, some caching technique may also speed up the time needed to generate a sentence, but it would certainly require a stable Internet connection, and the performance (whether we can build a real time application) would be heavily depends on the network latency.

- or we can build a standalone application on the mobile, so the whole process are powered by the computation power on the device alone. In this way user can use the application at any time, anywhere with the same performance, but it certainly demands a bit more powerful hardware to get the application running, which set the bar for hardware requirement, that could be another cost factor to consider.

At the end I have decided to build the application on mobile basis, based on the consideration that most useful application are more likely to proceed on a mobile device, as most of the picture taken and processed, shared today are from our mobile devices. To be able to use the model without a Internet connection would definitely border the possible use cases of our model, and also consider that machine learning on mobile device would be quite a new and interesting area to explore.

5.2.2 Building the application

Although much more machine learning are developed everyday now, it is still not so easy to bring it into a mobile application form, especially the high demand specification (of CPU and GPU computation power) has set the bar for what mobile can do and what mobile can't do, we will discuss more in the follow analyst part. After looking into existing project people have done, I have decided to build the mobile application on android platform, as it has much wider range of available hardware to choose from, and also considered that the available library are more mature and simple to use today, since tensorflow has a smoother connection with android platform device.

To start building the mobile application, first we need to prepare the tensorflow model file, from the trained file, it has all the values of variables saved into a checkpoint file (model.ckpt), we will transform the file into a standalone file so we can put it in a android application, along with the word dictionary file.

Then is to freezing the graph, which is turning the variables of the checkpoint file to constant ops that contain the variables, it will gives us a standalone file (.pb) so we can put it in android device and used later. The redundant operations like debugging, checkpoint logics are removed so we can produce a optimized (both in terms of performance and size) file to run on later.

Tensorflow has provided libraries for running project on android(through JAVA api), after we created the empty project, we can link the .jar libraries file in the /lib/ folder in the android studio file structure. We then also put the above standalone file into the assets folder, we can now use the trained model in android platform through the libraries provided interface. The interface is really easy and friendly to use, by only the following code, we already successfully initiated the tensorflow session on the android platform, it has gather all the resources files we need(the standalone and the word dictionary file).

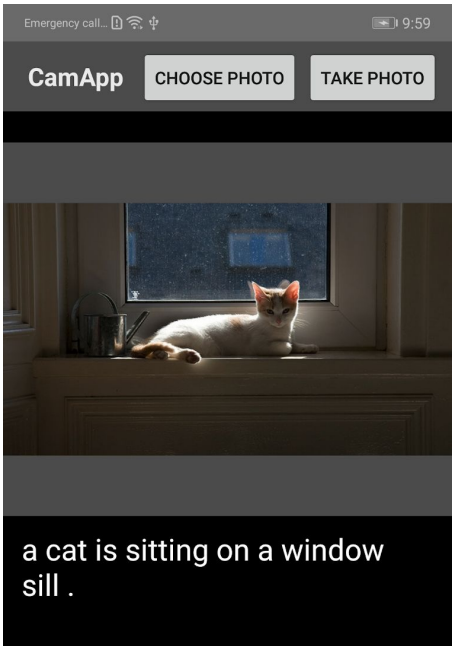
```
TensorFlowInferenceInterface InitSession(){
        inferenceInterface = new TensorFlowInferenceInterface();
        inferenceInterface.initializeTensorFlow(this.getAssets(), MODEL_FILE);
        OutputNodes = LoadFile(OUTPUT_NODES);
        WORD_MAP = LoadFile("idmap");
        return inferenceInterface;
    }
```

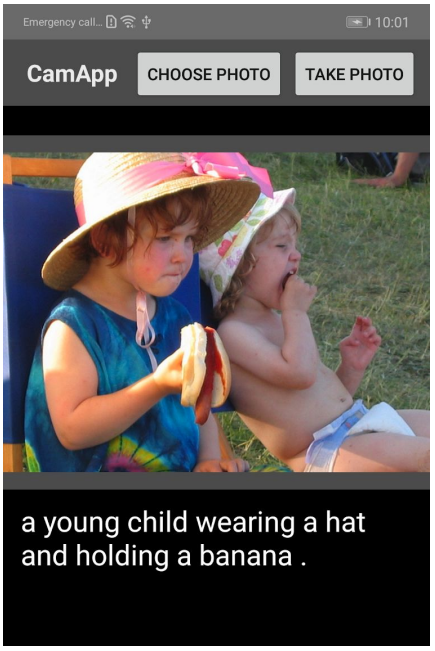
5.2.3 Analyst: Memory requirement

Although the android studio can compile the script to application that works on any android platform, however the freeze graph model does need to take a certain size in memory(198.6MB), this has become the largest hindrance of actually running the application on a mobile hardware. Then when we need to initialize the tensorflow session on the mobile device, it also add up the loads for device memory, it turns out that we need at least 4GB of main memory for the application to run normally, this requirement has seriously limited what devices can run the model.

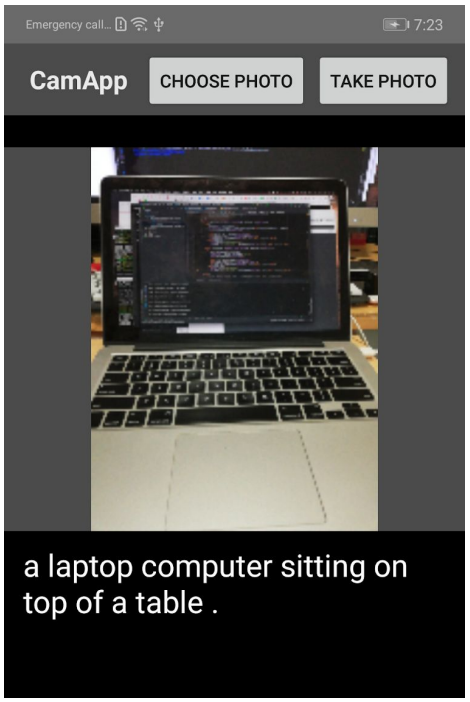
5.2.4 Analyst: Recognition accuracy of the model

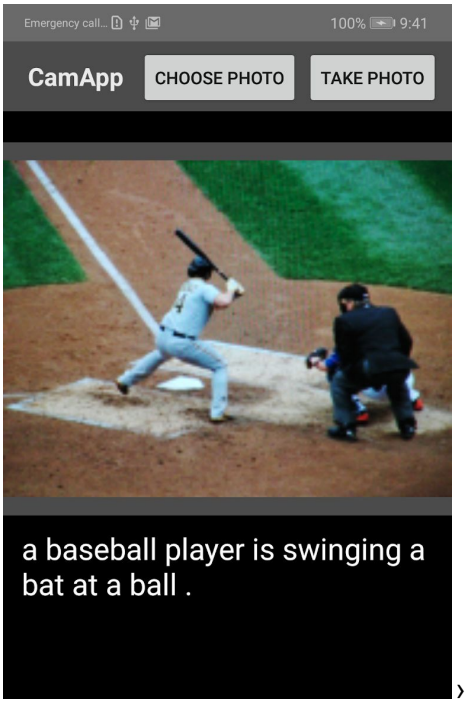
For the long time I have been study and working on the model in the remote machine on command line interface, so building a real world application would further help me to test the model performance when it really applies to real world scenario that beyond the test set.



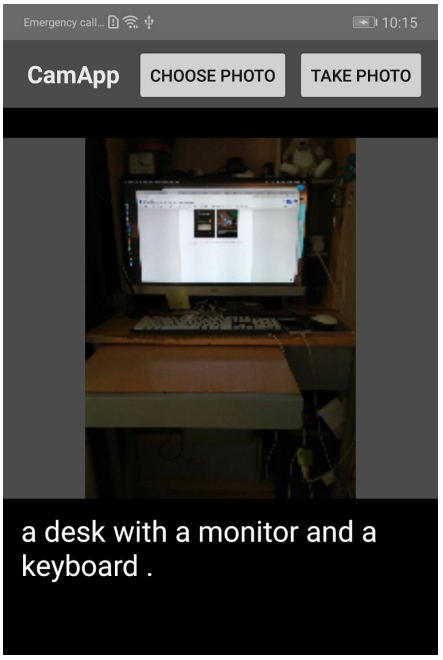


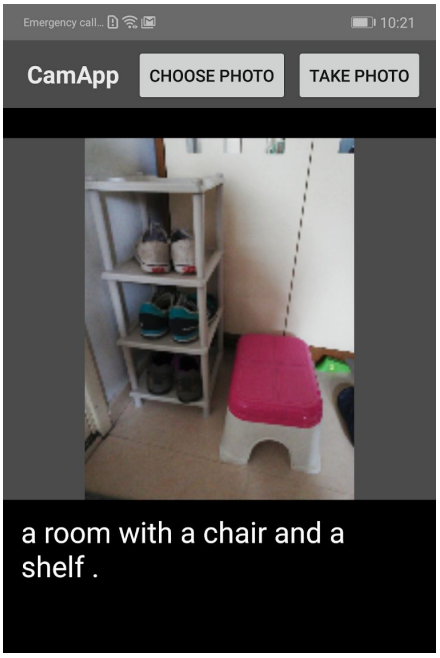
I have first conduct the testing on the MSCOCO dataset, the dataset we used for training, validation and testing, the result proven the statistical data I have analyst in semester one, we could observe some mistake caption the right image,maybe because the hotdog shape and color is very similar to a banana, (the features extracted matches to a banana).





Then I have also tested the model with some objects I found which never appear in the set of data we use, as what we expected the model works well to describe objects that has appeared in the training dataset, while having a hard time to recognize objects that it never sees, although it has tried to map the most likely match. We can also notice that the attention model picks the most significant object generate the description, so when there are mixed objects in an image, it will only choose to describe the most significant(size of portion/color contrast) parts. Our model takes in an image and preprocess it to 229x229 in size, so the decrease in resolution may also affects the recognition performance.





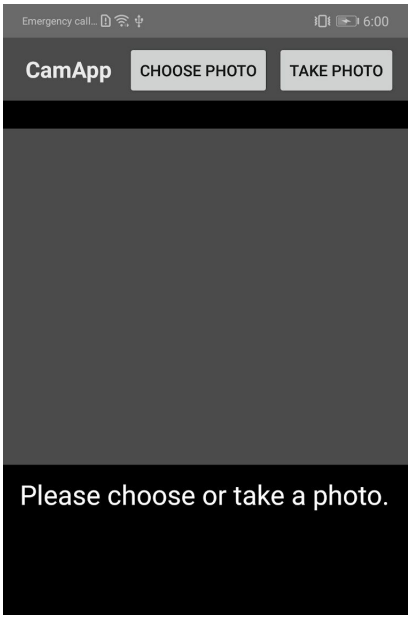
5.3 Approaches and Analyst

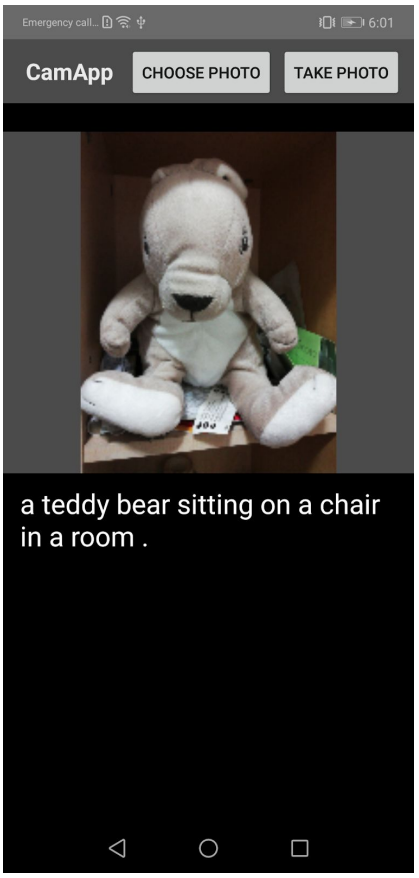
For the first goal of my implementation is try to run the most fundamental functionalities of my research model, which is to generate a descriptive sentence of a input image that contain daily objects. My first approach is to try making a real time application,

1. intake a image frame from camera source (at every certain time interval)
2. preprocess to fit the input parameters (e.g. resize and adjust the channel of the image)
3. initiate tensorflow session run the model to generate the caption

the result is not satisfying however, we hardly say we have achieved the real time translation as visually we do not have a instant generated sentence for every moment of our camera feeds, instead it gives sentence once in over ~4 seconds, also visually the camera is not doing a smooth recording most of the time. After timing each steps, obviously the problem is about the tensorflow session, it takes around ~3.5 to 4 seconds for initialize and running the model to get the returned caption, which is the majority of the process time. And the process power taken by initialize and running the session has eats up a significant part of the mobile computation resources.

After I have realised the hindrance of building a real time application, my second approach is to build a static caption generator, which I have replaced the live camera feeds to user input images every time(from choosing image from gallery or taking photo from camera in live time),



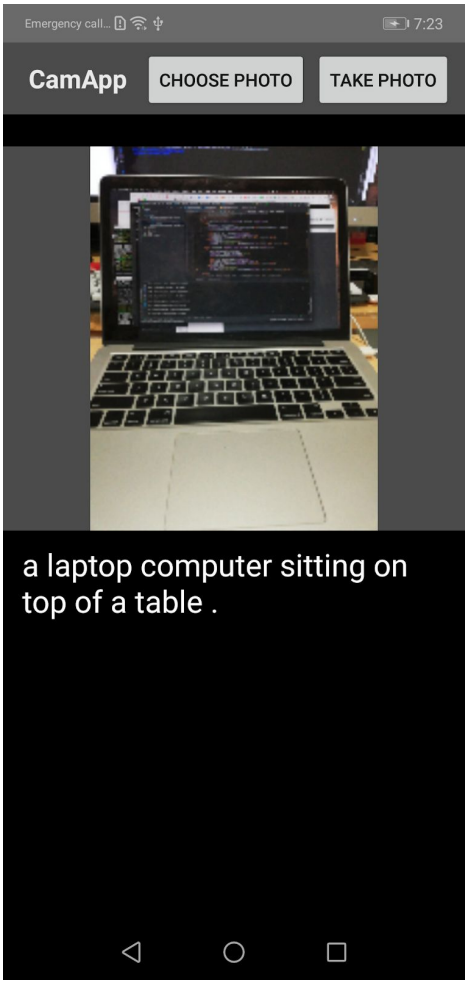


in terms of the the responsiveness, the result is quite satisfying, although the time taken for processing is comparable (also ~3.5 to 4 seconds), however the user experience is much

improved as it only idle when it is processing the image, the user can still feed an image from the camera, but without major frame dropping issue.

5.3.1 Taking it futher: (1) As an accessibility tool: Text to speech feature

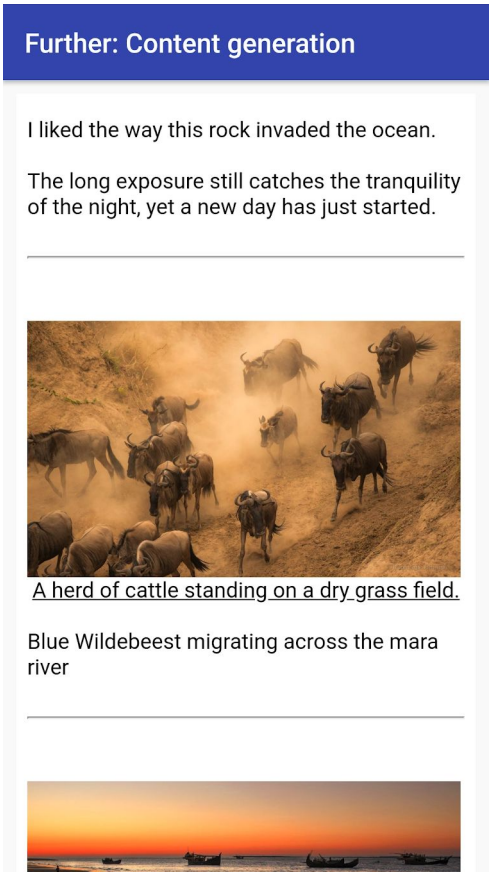
A pure caption generation application can hardly be useful in our daily life, so we will be explore some use cases towards our application. The first feature is probably adding a text to speech feature, so the mobile can read aloud the generated sentence, this has appears to be an accessibility feature in all smart phone, they can read out texts for visually impaired people, reading out loud the descriptive sentence would definitely helpful when they want a visual description of what going on.



we can clearly imagine that, a visually impaired person could identify many daily object, even of a distance without touching, when it is a real time application, they could plug an ear buds and keep a sense of what appears in front of them, it is both good for providing a visual sense and potentially ensuring their safety in daily life, as they could recognize if there may be dangers in front of them.

5.3.2 Taking it futher: (2) As an productivity tool: Automatic content generation for media business

Another potential application that would improve productivity would be a content generation application. It is a field called “Automated Journalism”, some media press has machine learning driven content, like New York Times, Reuters, BBC, The Guardian and Other Media Giants [29], they have implement machine learning for automated content at different levels. For experimenting the feasibility of applying our application to media business, I have try to extract an article from National Geographic, which is a photo content, each image has the original editor’s caption included.



Considered that our model was trained and specialized in recognition image of daily objects, I have select the images that our model would perform normally. We should always be aware that the model recognition power is always depends on the training value, although today there are scientist looking for approaches to generate caption include human-like artistic sense, but it is out of our project scope. As our model were implemented with supervised learning, it is work best with a specific field of data, for example, generating sentence describing some limited set of motion of football player, should provide us a satisfying accurate result. To take it even further, we could imagine this technology being used to generate descriptive sentence for live video, such as live news broadcast.

5.4 Possible Improvement

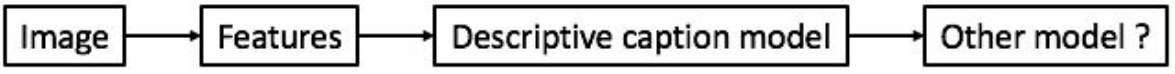
As in the implementation part, the most strict requirement of running our application is about the primary memory size of the device, the demands of memory(>=4GB) causes over half of the existing android device won't be able to run our application, we could imagine that it also implies what other machine learning model could run on a mobile.

If computation power is limited and chips standard remains as today, I would propose some possible solutions to run a large scale/heavy application on mobile device.

The first approach of course could be by using remote server to conduct the session initialization, it would be the most feasible solution for different machine learning application as remote machine (or cloud service) almost means unlimited resource today, it easy to acquire more resources and use it with pay per usage model. In our case, we could build a python server on a remote machine and the mobile can send and receive the data through http, but again it demands the user to stay connected to the Internet in order to use the service.

If running a standalone application on a mobile is a must (this could be mostly the case if we brought the model to much wider range of application in different fields), the second approach I imagine could be doing a better caching in between, we could save the process session to the storage by consider the device's memory limitation and to paging the process.

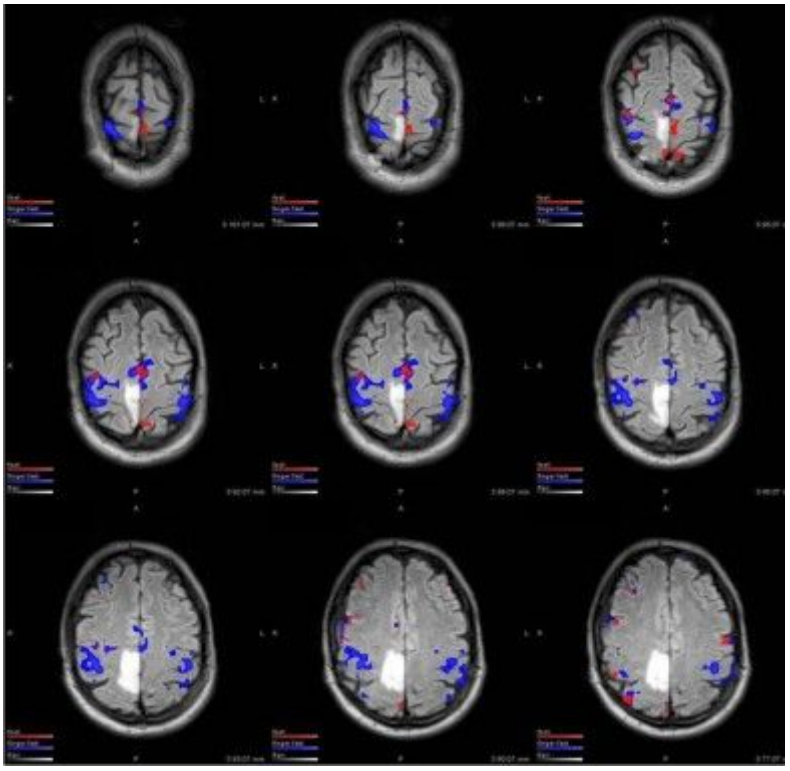
For building a more generic captioning model, beyond dailyobjects, and even applicable to different scenario, I can imagine that it could be done with distributed of layer for the model. For example we could obtain the first feature sets of the image to get an idea of what elements are inside, than apply the other layer to obtain other information of the image, like model specifically trained for recognizing object motion, color or texture(specialized for those specific objects).



5.5 Imagination and Prospects

The computer vision is still a fast growing fields, some successful application has been made and improving people's life, like monitoring, intelligence driven supermarket, marketing(locate people flows in a shopping mall for strategic planning), auto car driving. But there still lots of growing space that human could benefit from the image pattern recognition technology, for example,

- Medical use: SkinVision [30] uses ml image recognition, they can tell whether user may has skin cancer from mobile application taken image. We could imagine this could being great help in optimizing and improving the precision of some medical process, like generating a standard report of a computer scan for patient.



- Image search: Google Image provide image search features. We could imagine that if we utilize the caption generation model, we could match descriptive sentence to the set of images, user can input like "woman stand under a tree" and finally there will be precise image result and fits the user search parameters.

- Character Recognition: today carriers has utilize the power of computer vision to recognise handwritten address, there are also projects to conduct character recognition in translating printed material to digital form, the speed is extraordinary short and the precision is excellent when compare to the old OCR technique.

6. Conclusion

I have studied basics, history and the different approach of the image caption generating problem, by following the research of attention, I have also able to achieve a similar great performance as state from the research. During the time, I have learnt the advantage and uniqueness of attention that give better interpretability in the model generation process as it is closely related to human intuition. I hope this report can give a better understanding to our readers on the problem and solution proposed by the research. Finally, I would expect that similar encoder-decoder (translation with CNN and LSTM) approach like our project would be useful for tackle many other problems in different domains of machine learning.

After the further research and implement of application, I know much more about what can the model do, and what are the potential impacts of the model to the tomorrow's world. It is important for us to notice that, this is not only a model for generating a descriptive sentence for a image that contain dailyobjects, the model itself, is a generic machine translation model that could "translate data". Once we have two set of data, they could be in any different forms, if there are relation between them for us to find a pattern, once we gather sufficient information, we could give confident translation between that sets of data in many situation. This would be a long way for us to keep explore and develop application that could being great helps to humanities.

7. References

[1] Kyunghyun Cho. Introduction to Neural Machine Translation with GPUs. Nvida Developer Blog. May 2015.

[2] Rensink, Ronald A. The dynamic representation of scenes. Visual cognition, 7(1-3):17–42, 2000.

[3] Corbetta, Maurizio and Shulman, Gordon L. Control of goaldirected and stimulus-driven attention in the brain. Nature reviews neuroscience, 3(3):201–215, 2002.

[4] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. arXiv:1502.03044v3. April 2016.

[5] Kulkarni, G., Premraj, V., Dhar, S., Li, S., Choi, Y., Berg, A. C., & Berg, T. L. (2011). Baby talk: Understanding and generating simple image descriptions. In IEEE Conference on Computer Vision and Pattern Recognition.

[6] Fang, H., Gupta, S., Iandola, F., Srivastava, R., Deng, L., Doll´ar, P., Gao, J., He, X., Mitchell, M., Platt, J., Zitnick, C. L., & Zweig, G. (2015). From captions to visual concepts and back. In IEEE Conference on Computer Vision and Pattern Recognition.

[7] Yang, Y., Teo, C. L., Daume, III, H., & Aloimonos, Y. (2011). Corpus-guided sentence generation of natural images. In Conference on Empirical Methods in Natural Language Processing.

[8] Ortiz, L. M. G., Wolff, C., & Lapata, M. (2015). Learning to Interpret and Describe Abstract Scenes. In Conference of the North American Chapter of the Association of Computational Linguistics.

[9] Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey. ImageNet classification with deep convolutional neural networks. In NIPS. 2012.

[10] Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, Berg, Alexander C., and Fei-Fei, Li. ImageNet Large Scale Visual Recognition Challenge, 2014.

[11] Kiros, Ryan, Salahutdinov, Ruslan, and Zemel, Richard. Multimodal neural language models. In International Conference on Machine Learning, pp. 595–603, 2014a.

[12] Mao, Junhua, Xu, Wei, Yang, Yi, Wang, Jiang, and Yuille, Alan. Deep captioning with multimodal recurrent neural networks (m-rnn). arXiv:1412.6632, December 2014.

[13] Vinyals, Oriol, Toshev, Alexander, Bengio, Samy, and Erhan, Dumitru. Show and tell: A neural image caption generator. arXiv:1411.4555, November 2014.

[14] Donahue, Jeff, Hendrikcs, Lisa Anne, Guadarrama, Segio, Rohrbach, Marcus, Venugopalan, Subhashini, Saenko, Kate, and Darrell, Trevor. Long-term recurrent convolutional networks for visual recognition and description. arXiv:1411.4389v2, November 2014.

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

[16] Ba, Jimmy Lei, Mnih, Volodymyr, and Kavukcuoglu, Koray. Multiple object recognition with visual attention. arXiv:1412.7755, December 2014.

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

[18] Stanford University. CS231n: Convolutional Neural Networks for Visual Recognition. 2017.

[19] Hochreiter, S. and Schmidhuber, J. Long short-term memory. Neural Computation, 9(8):1735–1780, 1997.
[20] Zaremba, Wojciech, Sutskever, Ilya, and Vinyals, Oriol. Recurrent neural network regularization. arXiv preprint arXiv:1409.2329, September 2014.
[21] Pascanu, Razvan, Gulcehre, Caglar, Cho, Kyunghyun, and Bengio, Yoshua. How to construct deep recurrent neural networks. In ICLR, 2014.
[22] Williams, Ronald J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3-4):229–256, 1992.
[23] Lin, Tsung-Yi, Maire, Michael, Belongie, Serge, Hays, James, Perona, Pietro, Ramanan, Deva, Dollar, Piotr, and Zitnick, ´ C Lawrence. Microsoft coco: Common objects in context. In ECCV, pp. 740–755. 2014.
[24] Denkowski, Michael and Lavie, Alon. Meteor universal: Language specific translation evaluation for any target language. In Proceedings of the EACL 2014 Workshop on Statistical Machine Translation, 2014.
[25] Chen, Xinlei and Zitnick, C Lawrence. Learning a recurrent visual representation for image caption generation. arXiv preprint arXiv:1411.5654, 2014
[26] Fang, Hao, Gupta, Saurabh, Iandola, Forrest, Srivastava, Rupesh, Deng, Li, Dollar, Piotr, Gao, Jianfeng, He, Xiaodong, Mitchell, ´ Margaret, Platt, John, et al. From captions to visual concepts and back. arXiv:1411.4952, November 2014.
[27] Karpathy, Andrej and Li, Fei-Fei. Deep visual-semantic alignments for generating image descriptions. arXiv:1412.2306, December 2014.
[28] Karpathy, https://github.com/karpathy/neuraltalk2
[29] Corinna Underwood, Automated Journalism – AI Applications at New York Times, Reuters, and Other Media Giants, January 2018, https://www.techemergence.com/automated-journalism-applications/
[30] SkinVision, https://www.skinvision.com/

Appendix

A. Sample output result from our model