MANTECH Publications
Journal of Artificial Intelligence, Machine Learning and Soft Computing
Volume 2, Issue 2, July-December 2017

Visual Awareness Kit

Alex Abraham[1], Nidin Boban[2], Bashal C[3], Abi Abahai T[4]
Mar Athanasius College of Engineering, Kothamangalam
Email: *alexalx@gmail.com[1], nidinboban1996@gmail.com[2], bashalc95@gmail.com[3], abytom@gmail.com[4]*

Abstract

The Visual Awareness Kit (VAK) is an artificial intelligent system that helps the visually impaired people in the society. The proposed system is made with an innovative idea to help the blind to navigate easily through obstacles. A blind man has so many limitations while walking through the street, to identify an object, to avoid obstacles in front of him/her etc. As mentioned, VAK helps the blind, by identifying the scenes in front of him/her and the distance to the nearest obstacle. The information includes what exactly the obstacle is and the distance between him/her with the obstacle. And this information is communicated to the person through a headset. All this is done using artificial neural networks. The VAK consists of an autofocus camera for live image capturing, A high performance Raspberry pi module trained with convolution neural network for processing and captioning the image input from the camera..

Keywords: *Visual Awareness Kit (VAK), Raspberry pi, artificial neural networks.*

INTRODUCTION

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 achieves 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 computation 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 the project, there is a 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 cs231npartner with a clipper. Attention helps us to determine the relationship between the objects.

FRAMEWORK

A. Extract Features

The input of the model is a single raw image and the outputis a caption y encoded as a sequence of 1-of-K encoded words.

y = {y1,….,yc}, yi is an element of 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 = { a1,….al} , aiis an element of 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 information about relationship between salient features and clusters in the image[1].

B. Caption Generator

The model uses 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[1]

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

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

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

Where, respectively it, ft, ct, ot, ht are the input, forget, memory, output and hidden state of the LSTM. The vector $^{t} \in RD$ represents the context vector, capturing the visual information related to a particular input location. $E \in Rm \times K$ is an embedding matrix. m and n is the embedding and LSTM dimensionality respectively. σ and are the logistic sigmoid activation and element-wise multiplication respectively. The model define a mechanism φ that computes ^{t} from annotation vectors$a_{i,i} = 1,...,L$ corresponding to the features extracted at different image locations. And ^{t} is are presentation of the relevant part of the image input at time t4. For each location i, the mechanism generates a positive weight αi that can be interpreted as the relative importance to give to location i in blending the αi'stogether[1]. The model compute the weight αi by attention model f_{att} for which the model use multilayer perceptron conditioned on the previous state h_{t-1}.

$e_{ti} = f_{att}(ai,ht-1)$

$\alpha_{ti} = (\exp(e_{ti})) / (\Sigma^{L}_{k=1} \exp e_{tk})$

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

$z_t = \varphi(a_i,\alpha_i)$

where φ is a function that returns a single vector given the set of annotation vectors and their corresponding weights. The initial memory state and hidden state of the LSTM are predicted by an average of the annotation vectors fed through two separate MLPs.

$c_o = f_{init,c}(1\ L\ \Sigma L\ i\ ai)$
$h_o = f_{init,h}(1\ L\ \Sigma L\ i\ ai)$

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

$p(y_t|a,y_1^{t-1}) \propto \exp(Lo(E_{yt-1} + L_h h_t + L_z\ ^{t}))$

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

C. .Loss Function

We use a word-wise cross entropy as the basic loss function l_0. Furthermore, to encourage the attention function to produce more expressive output, we define l1, l2 as the variace of αt 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 hyper parameters.

D 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.[1]. 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 soft max 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 evector corresponding to the object the word wants to describe, andpassthemintoRNN.Motivatedbythework as,wedesignthearchitectures below. {ai} 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 finit,h and finit,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{ai}. The weight alpha is computed from the combination of h0, representing information of current word, and each of{ai},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{ai}a teach step, which mean salpha should put more weights on the area of the next word. This alpha is exactly the attention we want. Here for finit,h,finit,c, we use multilayer perceptrons (MLP). For that, 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{ai}directly to the first input of LSTM0 through 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[2].

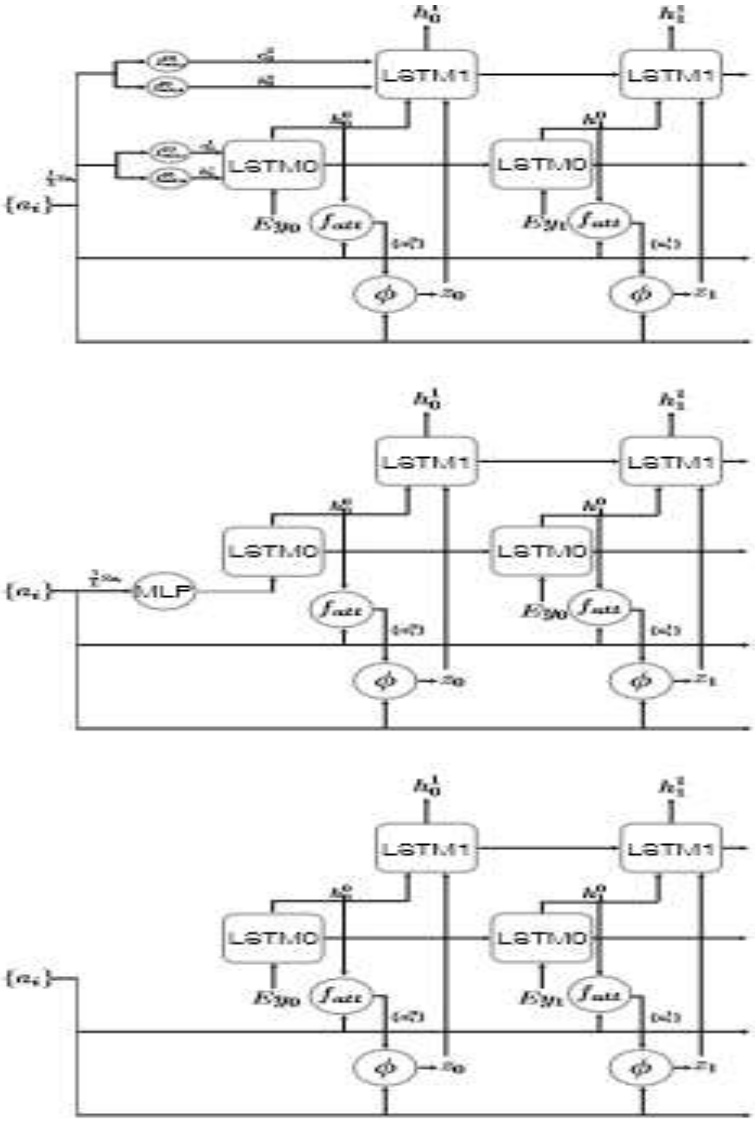

Figure 1: The first is original model. The second is Model with more weights on attention. The third is Model only depending on attention we generate a vocabulary from the sentences we need to set a threshold to decrease the classification error. The threshold should not only remove the spares words, 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[3].

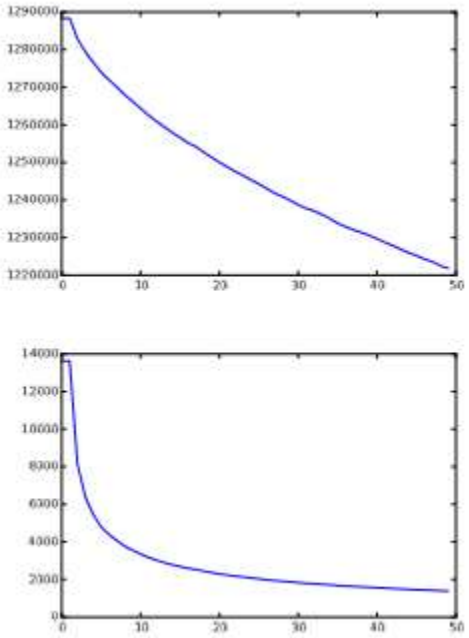

Figure 2: the upper figure is vocabulary size – threshold curve, the lower figure is total word count–threshold curve the latter one is linear, so we choose 10 as our threshold. For images, we preprocessing them by cropping them to 224x224 and subtract the mean. Because they are already a large number of data, we dont do any data augmentations. We tried to extract features of "inception3b", "inception4b" and "inception5b" from Google net. 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[3].

III DISTANCE CALCULATION

The Distance Identification Module identifies and alerts the user about the obstacles nearby. The module calculates and analyzes each object in every frame to give output to the user. JAVA has been primarly implemented in the module as a language of choice. Each frame from the video input will be written as files. And this files are read using the image class in java. These files are then processed by a objection detection algorithm.

A. Buffered Image Class

In the Java 2D API an image is typically a rectangular two-dimensional array of pixels, where each pixel represents the color at that position of the image and where the dimensions represent the horizontal extent (width) and vertical

extent (height) of the image as it is displayed.

The most important image class for representing such images is the java Aw image. Buffered Image class The Java 2D API stores the contents of such images in memory so that they can be directly accessed.

Applications can directly create a Buffered Image object or obtain an image from an external image format such as PNG or GIF.

In either case, the application can then draw on to image by using Java 2D API graphics calls. So, images are not limited to displaying photographic type images. Different objects such as line art, text, and other graphics and even other images can be drawn onto an image (as shown on the following images).

The Buffered Image class is a cornerstone of the Java 2D immediate-mode imaging API. It manages the image in memory and provides methods for storing, interpreting, and obtaining pixel data. Since Buffered Image is a subclass of Image it can be rendered by the Graphics and Graphics2D methods that accept an Image parameter[6].

A Buffered Image is essentially an Image with an accessible data buffer. It is therefore more efficient to work directly with Buffered Image. A Buffered Image has a Color Model and a Raster of image data. The Color Model provides a color interpretation of the image's pixel data.

B. Object detection algorithm (Simplified)

Step 1: Read the RGB values of each pixels in the file

Step 2: Convert the image to grayscale by dividing the sum of RGB values by

Step 3: Divide the image into 3 parts. The middle section or the more focused part and the part to its left and right.

Step 4: For each of these parts divide again into 3 part top, bottom and middle.

Step 5: Now we have 9 blocks. For each block from the center we compare a shade of the image to calculate the size of object embedded in that particular block.

Step 6: Now a comparison is made between these results to find out which of these blocks are related to each other this helps to identify large and close objects.

Step 7: From these comparison we find out the closest obstacle to the user and provide

an alert for the same.

C. Gray Scale conversion

In order to convert a color image to Grayscale image, you need to read pixels or data of the image using File and Image IO objects, and store the image in Buffered Image object.

Further, get the pixel value using method **getRGB()** and perform GrayScale() method on it.

```
Color c = new Color(image.getRGB(j, i));
int red = (c.getRed() * 0.299);
int green =(c.getGreen() * 0.587);
int blue = (c.getBlue() *0.114);
```

The method getRGB() takes row and column index as parameter. The last step is to add all these three values and set it again to the corresponding pixel value.

```
int sum = red+green+blue;

Color new Color = new Color(sum,sum,sum);

image.setRGB(j,i,new Color.get RGB());
```

CONCLUSION

The visual awareness kit is a AI aided solution for a blind person navigation problem, The kit simply guides the blind through the obstacles in front of them. It can alert them of obstacles and also identify them if needed. The system is implemented with a camera, and a small processing unit. A high performance computer with the support of GPU and good amount of RAM helps for the faster processing of images. Training more images makes the system more efficient. With some more additional components and support of GPU, a wearable system can be developed which makes blind to drive vehicles through the road.

ACKNOWLEDGMENT

We express our sincere gratitude and thanks to Dr. Soosan George T, our Principal and Dr. Surekha Mariam Varghese, Head Of the Department, our project guide Computer Science & Engineering, for providing the facilities and all the encouragement and support.

REFERENCES

1. Scalable and Secure Sharing of Personal Health Records in Cloud Computing UsingAttribute-Based Encryption. Ming Li, Member, IEEE, Shucheng Yu, Member, IEEE, Yao Zheng, Student Member, IEEE, Kui Ren, Senior Member, IEEE, and

Wenjing Lou, IEEE transaction on parallel and distributed systems

2. M. Li, S. Yu, N. Cao, and W. Lou, "Authorized Private KeywordSearch over Encrypted Personal Health Records in CloudComputing," Proc. 31st Int'l Conf. Distributed Computing Systems(ICDCS '11), June 2011

3. Szegedy C, Liu W, Jia Y, et al. Going deeper with con-volutions[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 cap- tion generator." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156-
5. 3164. 2015.

6. Donahue, Jeffrey, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venu- gopalan, 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.

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

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

9. Mnih V, Heess N, Graves A. Recurrent models of vi- sual attention[C]//Advances in Neural Information Pro- cessing Systems. 2014: 2204-2212.

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

11. Hermann K M, Kocisky T, Grefenstette E, et al. Teach- ing machines to read and

comprehend[C]//Advances in Neural Information Processing Systems. 2015: 1684-1692.