Eye of Aurora Final Report

Zeqi Li, Weixin Wang, Haohang Yan, Yuankai Huang
Advisor: Pramod Khargonekar
Department of Engineering
University of California, Irvine
Irvine, CA, USA
{zeqil1 & weixinw1 & haohangy & yuankaih }@uci.edu

***Abstract* - Visually impairment is an essential issue in the world that many people are trying to solve. Our project aims to help visually impaired people know what is happening around them. Therefore, we made a pair of camera glasses that can describe the scene in front of the user with sound. The describing sentences are generated by deep learning. We trained our own image caption models and found that VGG 19 as the neural network model with Flickr8k as the dataset performed best. The hardware includes a camera, an earphone, a glass frame, an LCD touch screen, batteries, and a Raspberry Pi 4. With our project, visually impaired people can truly "see" and engage with the world around them.**

Index Terms - image caption, text-to-speech, visually impaired, Artificial Intelligence.

I. Introduction

According to the 2020 report of the World Health Organization, there are about 253 million people with visual impairment in the world [1]. Therefore, our project goal is to help visually impaired people "see" the world. We made a pair of camera glasses that can describe the scene in front of the user with sound.

For years, scientists have been trying to develop assistive tools that make visually impaired people more independent and aware of their surroundings [2, 3, 4]. Researchers have made many attempts, including but not limited to voice assistance, ultrasonic assistance, camera-based assistance, biomedical technology, voice-based navigation systems [5]. There have been a lot of developed navigation systems for visually impaired people. However, most of the main ideas of these systems are merely to make people aware of the obstacles in the path or to suggest a safe path (if available) to the left or right side [4, 5]. Although blind guidance and obstacle avoidance tools are increasingly developed, visually impaired people are vulnerable to self-abasement, confusion, and anger because they ultimately cannot see and integrate into the world around them [1]. Hence, our project focuses less on navigation, but more on making the user understand what's going on around them.

In our scenario, when the user clicks on the touch screen, the camera attached to the glasses will take a picture and transmit it to a microprocessor. Then, our Image Caption model implemented with TensorFlow will take the picture as input to produce a natural sentence that can describe the main content of the picture. After that, the text-to-speech program based on eSpeak will speak out that sentence. Finally, the picture and sentence will be displayed on the screen for developing and testing. We chose the Raspberry Pi 4 to serve as the microprocessor.

For text-to-speech, we used eSpeak to serve as the software speech synthesizer and generate speech. It performs well on our Raspberry Pi 4. For Image Caption, we used Tensorflow as our framework. We have tried InceptionV3, VGG 16, VGG 19, and MobilenetV2 as our neural network models, and have tried Flickr8k and Flickr30k as our training datasets. Among all models and datasets, we use VGG 19 with Flickr8k performing best. We used BLEU (BiLingual Evaluation Understudy) score to evaluate the performance of our models. A detailed table comparing the performance of different models and datasets is provided in Section IV. A typical result of Image Caption is shown in Fig 1. Fig 2 shows that we were testing the function of Image Caption on our project.

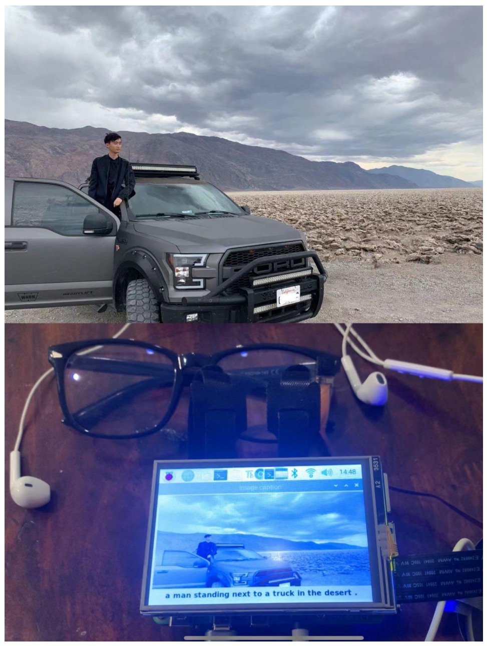

Fig. 1. A typical result of Image Caption. The sentence on the top, "a man standing next to a truck in the desert", is describing the photo.

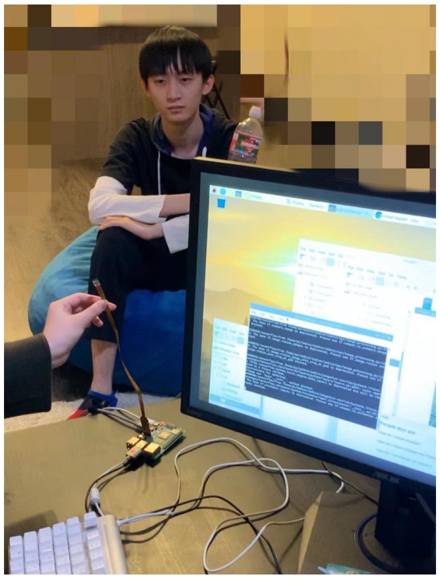

Fig. 2. We were testing the project.

II. Material Used

For materials, we used a spy camera to take pictures for our image inputs and a Raspberry Pi for the processor. We used earphones as our audio output and an LCD touch screen as our image output. For the power supply, we are using a portable type-18650 battery that can charge Raspberry Pi. When we touch the LCD screen, a signal will be sent to the program. Whenever the screen is touched, we will process the image and output the description through the earphone and the image and description on the screen. Fig 3 has shown the materials and the product we have designed.

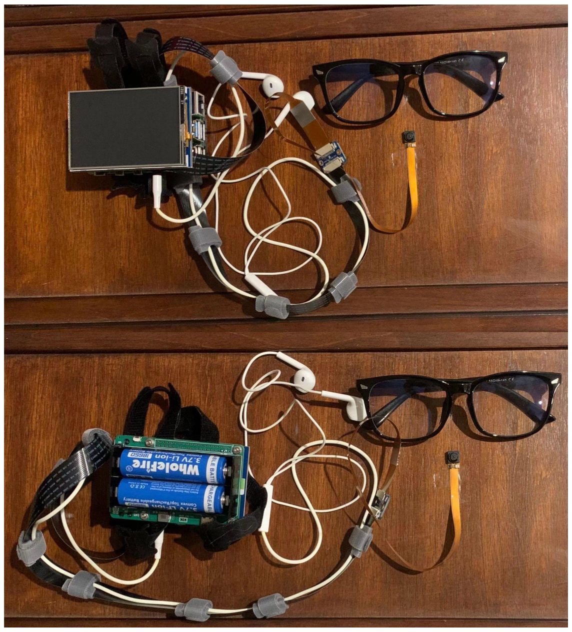

Fig. 3. All the hardware used.

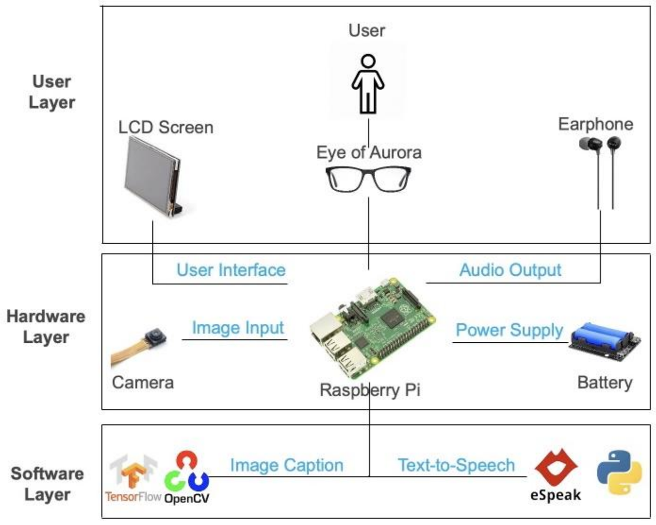

Fig. 4. The high-level design of the project.

Fig 4 shows the high-level design of our project. The hardware layer consists of six parts: A camera that can take pictures. An LCD screen that triggers the software program and shows the image we captured, and the camera. An earphone that can deliver the sound caption to the users. A Raspberry Pi that serves as the microprocessor to run Image Caption and text-to-speech. A portable battery that powers the Raspberry Pi, especially when we need to take our product outdoors. A glasses frame that the user can wear. The LCD screen, camera, earphone, and battery are connected to the Raspberry Pi. The front end of the camera is attached to the glasses frame. In our scenario, the user will wear the glasses frame and the Raspberry Pi and its attachments will be strapped to the user's arm. When the user touches the LCS screen, the camera in the glasses will take a picture and transmit it to Raspberry Pi. The Raspberry Pi will run an Image Caption program to convert the photo into a text description, which is then played back by voice by text-to-speech technology. Inside the image caption program, a neural networks-based TensorFlow framework is used to do the prediction for images. And after neural networks finish predictions, the eSpeak library will convert predictions to audio output.

IV. Method

A. IMAGE CAPTION

1) Preparation:

Considering our hardware limitations, we choose the Flickr8k and Flickr30k as our datasets. These datasets are relatively small but also are reliable datasets. Meanwhile, we also have the glove300d word vector as our embedding vector.

2) Preprocess:

For the parts of the image, we decided to use an existing model to do the feature extraction and no need to spend lots of time on training on images. For the text parts, we are going to use sequences to store the information. First, we have several sentences of descriptions for each image. We add “startseq” to the beginning of each sentence as the start of the sequence and “endseq” as the end of the sequence. Meanwhile, every sentence needs to process to be the form in table II. After finishing all sequence creation, we create a vocabulary that has 8173 words using a dictionary to project every word into an integer.

TABLE I
A SAMPLE PREPROCESS OF TEXT SEQUENCE

X1(image vector)	X2(Text sequence)	Y(next word)
4096-dimension vector	“startseq”	a
4096-dimension vector	“startseq, a”	boy
4096-dimension vector	“startseq, a, boy”	is
4096-dimension vector	“startseq, a, boy, is”	running
4096-dimension vector	“startseq, a, boy, is, running”	on
4096-dimension vector	“startseq, a, boy, is, running on”	grass
4096-dimension vector	“startseq, a, boy, is, running, on, grass”	endseq
4096-dimension vector	“startseq, a, boy, is, running on, grass, endseq”	

3)Model:

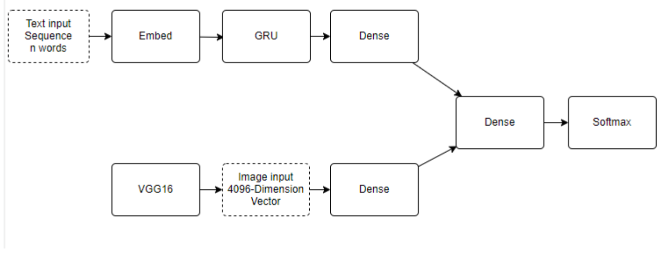
Text input Sequence n words
Embed
GRU
Dense
Dense
Softmax
VGG16
Image input 4096-Dimension Vector
Dense

Fig. 5. Model architecture.

In general, we are going to merge architecture [6] as our architecture. In our model, after comparing several models, we choose VGG16 as our image encoder. Normally, VGG16 will have a classification layer using softmax as an activation function to output the predictions. In our model which is illustrated in figure 5, we remove this layer and only use the fully connected layer storing 4096 dimension vectors as image inputs to the caption model. For the text parts, we first use the glove word vector to embed our text input sequence. After that, we use the GRU model to extract features from input text vectors that have 256 dimensions. And then, we combine the image features and text features together using a dense layer to process. Finally, we use the softmax layer to output predictions. Because our vocabulary is large, we need to use softmax as a multiple classification function which is illustrated in figure 6.

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \text{for } i = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K.$$

Fig. 6. Softmax function.

4)Predictions:

Because the output of our model is an integer, we use our vocabulary to get the real word that we want. After that, we need to use new word sequences and image vectors as our new input to the neural network model to do prediction again. When we see the word “endseq” in our sequence, we will stop the iteration and output a sentence as our final prediction.

B. *Text to Speech*

We want our description of the image from the image caption function to be played out to the users. Therefore We implement a text-to-speech function using eSpeak. eSpeak is an open-source synthesis method that can play our sentences in sounds. It determined the pronunciation of the words in a sentence. When a word is put into the eSpeak system, its phrasing, intonation, and duration will be analyzed and the speech of the word will be produced. Since the output for image caption is a series of words. We combined words into strings and outputted results through audio by eSpeak in the ‘os.system’.

V. Result and Performance

1) Model choose:

TABLE II
TRAINING EVALUATION OF DIFFERENT MODELS

datasets	Flickr8k			
model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
VGG16+GRU	0.540525	0.293896	0.199078	0.08958
Inception V3+GRU	0.451857	0.196511	0.121756	0.045565
MobilenetV2 +LSTM	0.49615	0.264625	0.182899	0.085912
VGG19+GRU	0.504208	0.286042	0.209405	0.10736

Table II shows the evaluations of our choosing models. In the dataset, we used BLEU [7] as our measurement tool because the traditional accuracy measurement is not compatible with the image caption function. Meanwhile, BLEU [7] is a measurement that can evaluate the performance of machine translation and is prevailingly accepted for the evaluation of a certain model. Therefore, BLEU becomes our choice of evaluation of models. The table above shows that the results of our model are not high enough to completely accomplish image captions.

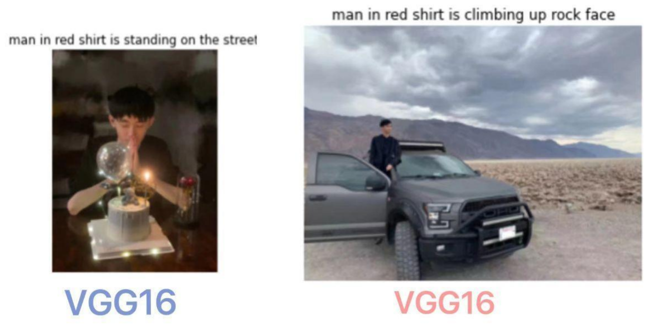
man in red shirt is standing on the street
man in red shirt is climbing up rock face
VGG16
VGG16

Fig. 7. VGG16

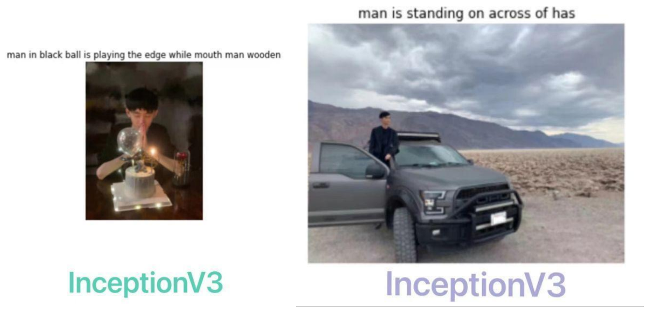
man in black ball is playing the edge while mouth man wooden
man is standing on across of has
InceptionV3
InceptionV3

Fig. 8. InceptionV3

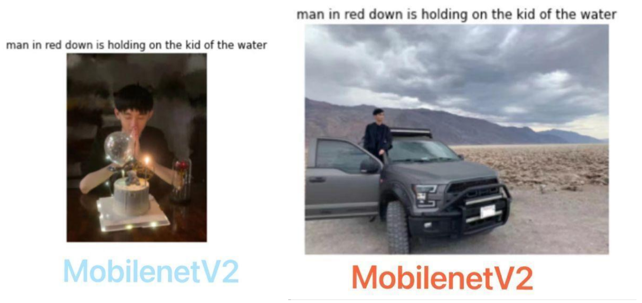
man in red down is holding on the kid of the water
man in red down is holding on the kid of the water
MobilenetV2
MobilenetV2

Fig. 9. MobilenetV2

In figure 7, 8, and 9, we compare the actual results of three different models. As the figures show, VGG16 has the best description of the images even though it doesn't completely describe the picture. However, in inceptionV3 and MobilenetV2. The descriptions are confusing and even are not complete sentences.

2) Hyperparameter experiment:

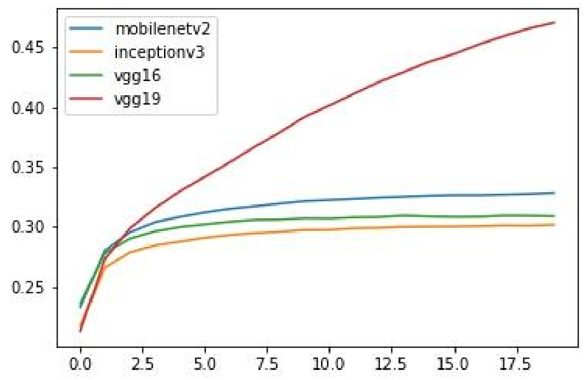

Fig. 10. Accuracy of several models

After comparing the results of different models, we finally choose vgg19 as our final mode. Then we start several experiments to get better results. At first, we found out that the accuracy of our model was very low and converged very quickly. Therefore, we decreased the learning rate from 0.001 to 0.0001. Meanwhile, in order to learn more about the image and text information. We used 0.3 dropout layer instead of 0.5 dropout layer. The accuracy was significantly increased after changing the hyperparameters which is shown in figure 10.

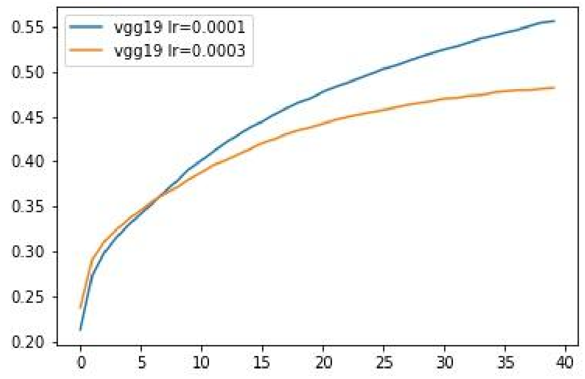

Fig. 11. Different learning rates

Also, the accuracy doesn't meet our expectations. Hence, we did several experiments to test the accuracy. As shown in Fig 10 above, the accuracy of VGG19 isn't converging currently. To make the accuracy higher, we use large epochs to do the experiment. In the meantime, we also tried another learning rate 0.0003 to testify which one is better. As figure 11 shows above, the learning rate with 0.0003 starts with higher accuracy but lower accuracy at last. However, the learning rate with 0.0001 has higher accuracy and can easily expect higher accuracy if we use larger epochs. Therefore, our final experiment is based on a learning rate = 0.0001 and with larger epochs.

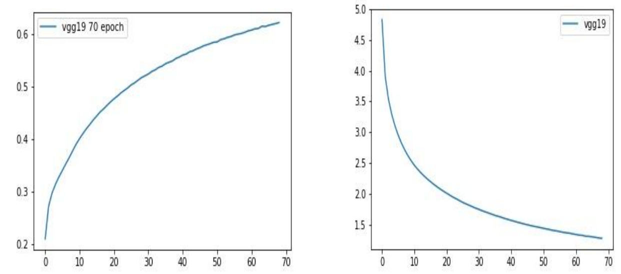

Fig. 12. Final model accuracy and loss

As shown in figure 12, our accuracy reached about 0.65 and loss reached about 1.27 at 70 epoch. Although our final accuracy is still not as high as other models like google model img2txt, we have a relatively small difference with google's model in BLEU scores in table III.

TABLE III
COMPARISON WITH GOOGLE'S MODEL

Datasets	Flickr8k			
Evaluation	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Our model	0.505158	0.278898	0.199664	0.103846
Google Img2txt	0.561677	0.319980	0.223410	0.114370

VI. Summary and Conclusion

Eye of Aurora is a pair of camera glasses that can describe what it sees in plain English. It contributes to several of the NAE grand challenge problem areas including health, learning, the joy of living, and accessibility (for disabilities). Allowing visually impaired people to see and engage with the world would bring them joy, reduce their anxiety and depression, and help them with learning. Accessibility is the key design concept in our project.

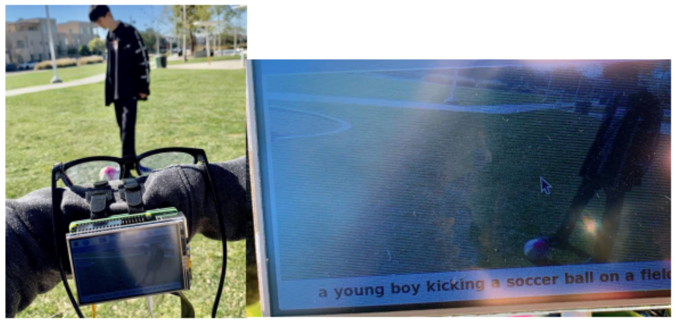

Fig. 13. A field test of Eye of Aurora

Fig 13 shows a field test of Eye of Aurora. It generated "a young boy kicking a soccer ball on a field" to describe what it "saw," which is correct. Visually impaired people might be used to hearing "Obstacle in the front. Please turn left." With our project, they will hear "five children playing soccer on the playground", "a dolphin leaping out of the sea", and "a meteor streaking across the sky."

VII. Acknowledgement

We thank Professor Pramod Khargonekar for his valuable advice. We thank Professor Stuart Kleinfelder for being our instructor in EECS 159A/B and his Shark Tank funding. Many equipment sources are from the funding. We thank Sahar Aali, Floranne Ellington, and Xing Xia for being our TAs. We thank Arnav Malawade for the useful discussion. We thank all group members: Zeqi Li, Weixin Wang, Haohang Yan, and Yuankai Huang.

References

[1] Demmin DL, Silverstein SM. Visual Impairment and Mental Health: Unmet Needs and Treatment Options. Clin Ophthalmol. 2020;14:4229-4251. Published 2020 Dec 3. doi:10.2147/OPTH.S258783
[2] Liu OC.A., Li SK., Yan LQ., Ng SC., Kwok CP. (2020) A Visually Impaired Assistant Using Neural Network and Image Recognition with Physical Navigation. In: Han M., Qin S., Zhang N. (eds) Advances in Neural Networks – ISNN 2020. ISNN 2020. Lecture Notes in Computer Science, vol 12557. Springer, Cham. https://doi.org/10.1007/978-3-030-64221-1_24
[3] Hengle, A. Kulkarni, N. Bavadekar, N. Kulkarni and R. Udyawar, "Smart Cap: A Deep Learning and IoT Based Assistant for the Visually Impaired," 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 2020, pp. 1109-1116, doi: 10.1109/ICSSIT48917.2020.9214140.
[4] Nadia Kanwal, Erkan Bostanci, Keith Currie, Adrian F. Clark, "A Navigation System for the Visually Impaired: A Fusion of Vision and Depth Sensor", Applied Bionics and Biomechanics, vol. 2015, Article ID 479857, 16 pages, 2015. https://doi.org/10.1155/2015/479857
[5] C. K. Lakde and P. S. Prasad, "Navigation system for visually impaired people," 2015 International Conference on Computation of Power, Energy, Information and Communication (ICCPEIC), 2015, pp. 0093-0098, doi: 10.1109/ICCPEIC.2015.7259447.
[6] Tanti, M., Gatt, A. and Camilleri, K.P., 2018. Where to put the image in an image caption generator. Natural Language Engineering, 24(3), pp.467-489.
[7] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu. BLEU: A method for automatic evaluation of machine translation. In ACL, 2002.

Appendix1

The technical standards that are relevant to our project include:

1. IEEE 1625-2004 - IEEE Standard for Rechargeable Batteries for Portable Computing
2. HDMI, which implements the EIA/CEA-861 standards.

3. Released in 1996, the USB standard is maintained by the USB Implementers Forum (USB-IF).
4. IEEE 802.11-2016 - IEEE Standard for Information technology - Telecommunications and information exchange between systems Local and metropolitan area networks--Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications

We choose these technical standards not only to effectively avoid the emergence of incompatible products and technologies wasted time, but also to ensure and improve the performance and quality of development, shorten the development cycle, save development funds, save manpower and material resources, etc. For example, we use the HDMI standard to reduce the impedance of the circuit to optimize the signal attenuation and mutual interference between signals. Our design complaints multiple technical standards including USB and WIFI. For instance, we choose the USB interface in accordance with the standards described in the USB transmission speed, maximum power carrying capacity and use it to select the most suitable usb cable for our experiments. We also strictly follow the usb cable connection standards to complete the alignment in our circuit.

Appendix2

A. *Challenge 1*

At first, we trained our model and found that our accuracy is significantly low, which is about 30%. Then we are confused about whether our model is implemented correctly. And then we tried different models like mobilenetv2, vgg16 to see the results. Still, the accuracy is about 30% and has no improvement. Therefore, we were starting to think about our architecture. After looking up several articles, we were sure about the feasibility of our model and found that the evaluation measurement for image caption models is different from traditional image classification. We found that BLEU [4] is used in most papers about image captioning. Hence, we tried BLEU to evaluate our model and finally get a relatively acceptable result for our model.

B. *Challenge 2*

Another challenge we have encountered is that we are not familiar with the Linux-based Raspberry Pi system. We spent a lot of time learning about it and now we can use Linux commands proficiently to carry out software development faster. In implementing OpenCV and different versions of TensorFlow, we were having trouble compiling the file we created and edited. We tried different versions of TensorFlow to run different libraries among various models we tested. Finally, we have selected the correct model and were able to run it in a Linux environment.

C. *Challenge 3*

As students, our budget is very tight. Hence, cost control is also a big challenge for us. Of all our hardware, microprocessors are the most expensive and the most difficult to choose. Microprocessors that are too cheap may not meet our project requirements, while microprocessors that are too expensive may waste performance and money. After many comparisons, we believe that Raspberry Pi 4 is the most cost-effective and meets the needs of our project. However, as RAM (random access memory) increases, the price of the Raspberry PI 4 can range from $35 to more than $200. In order to save costs, we decided to use our own computer GPU to train the model so that we could use the cheaper Raspberry Pi 4 to achieve our requirements. Finally, we applied for Shark Tank funding to solve the budget problem.

Appendix3

For both software and hardware security issues, our group has considered various factors before and during the experiment to prevent potential security problems. For hardware safety, we considered the possibility that the main hardware, Raspberry Pi 4, could be damaged or lost by the contact of other people besides the group members, so we decided to arrange the experimental site in a small, safe room, which was protected by the group members for each experiment. To prevent the hardware from overheating and causing safety situations such as lithium battery explosion and fire, we kept the hardware away from heat sources and enhanced heat dissipation. Our experiments cannot be conducted without electricity, so to prevent electrocution and electrical leakage, we check the circuit connections before turning on the power for each experiment. We also prohibit eating and drinking during the experiment to prevent leakage and short circuit from spilling liquid on the equipment.

Developing software is complex and systematic work, which contains a large number of limiting factors. There are always security holes in the computer software. Therefore, we will open the system firewall when we download resources online, to isolate the internal and external networks and establish the security gateway to prevent hacking to the maximum extent. We always ensure that programming specifications are followed when programming with Tensorflow. We keep track of our activities on our Raspberry Pi and record our important moves. We evaluate and check the version of our library biweekly to maintain a secure status. To strengthen our security level, we avoid letting people who are not in our team use the operating system.