

# “Alexa! What do you See?”

*A Voice Activated Automated Image Captioning System on VizWiz-Captions Dataset*

**Padmavati Sridhar    Shaji Kujumohamed    Shwetha Chitta Nagaraj**

University of California, Berkeley

August 2020

## Abstract

Automated image captioning has seen a sea of new research and development in the last 6 years with advanced techniques in vision computing and language development. Bringing this paradigm into real world settings and making an impact with real use cases such as helping the visually impaired caption objects and scenes in their natural surroundings is the objective of our image captioning system. We use the VizWiz-Captions dataset which has images sourced from the visually impaired and captioned with details to serve their needs, to train an image captioning model based on the Show, Attend and Tell architecture in IBM Cloud. By employing the power of edge computing on the trained model, we create a system where a user can talk to an Amazon Echo device and ask “*Alexa! What do you See?*” by pointing a cellphone camera towards an object or scene that needs to be captioned. The edge device performs inference on the image and sends the caption back to Alexa in under 5 seconds to communicate the text back to the user.

## 1. Introduction

Image captioning is one the most challenging tasks in computer vision which aims to automatically generate natural descriptions of images by detecting objects in an image but also to understand their interactions and then verbalize them using a natural language like English. However, the advancement in research in computer vision and natural language processing with state-of-the-art algorithms, has impregnated into several popular technology services like Facebook’s social media[1] and Microsoft’s Powerpoint[2] to help the visually impaired get sight into the images they encounter in such digital environments which they naturally cannot see. However, such

systems have been designed using large-scale publicly available datasets where images are scraped from the internet(Flickr) and captioned through crowdsourced workers. Such datasets don’t seem to serve the needs of the visually impaired as they have sought to seek descriptions from human powered services[3, 4, 5, 6]. But with our system the user group can get a faster and more private alternative within their environment using an image captioning model trained on a new publicly available dataset called **VizWiz-Captions**[11]. The images for the dataset are sourced from the visually impaired and captioned to suit their needs. The overall system architecture of our system is shown in Fig. 1. Amazon Echo devices have found home in nearly 70% of US homes who use voice activated AI assistants[7]. And have particularly

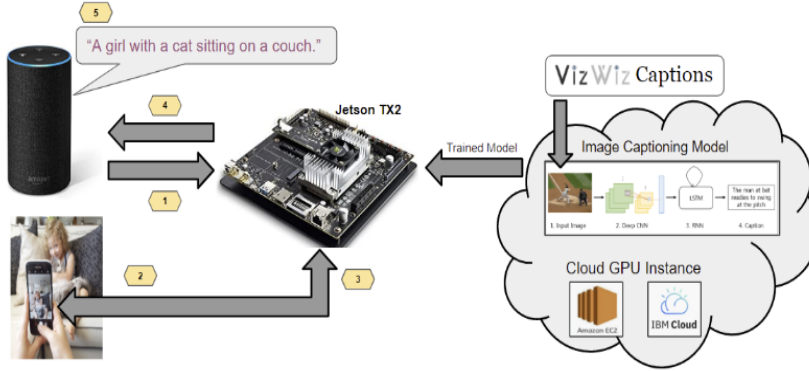


Fig 1: Overall Architecture

**Data Flow:**  
With trained image captioning model in Jetson TX2.

**User asks** "Alexa! What do you see?"

- 1 - Alexa sends a signal to Jetson to capture picture from camera
- 2,3 - Jetson captures image from camera

Jetson does inference using pretrained model and to generate caption text.

- 4 - This is sent back to Alexa
- 5 - Alexa says the text aloud to the user.

been useful to the visually impaired to help them with their daily needs[8] like getting the time, ordering groceries, reading books, playing music etc. We tried to use this popular home device where a visually impaired person can command the device to describe what they cannot see. They would just need their cell phone camera to shoot what they want described. The Echo device interacts with an edge computing AI device like NVIDIA's Jetson TX2[9] running within the user's home or private space to caption the image from the camera and send it back to the Echo device to say it aloud to the user. All this without sending details of their images or captions outside of their private space.

The captioning in the Jetson is enabled by a pre-trained image captioning model trained on the cloud[10] on VizWiz-Captions dataset. The model is based on encoder-decoder architecture for image captioning and uses an adaption of Show, Attend and Tell architecture[12].

With this system, a visually impaired person can get descriptions of objects or scenes within his/her surroundings in real time with a voice activated command and a cellphone camera.

Code is available at <https://github.com/shwethacn/W251-Final-Project>

## 2. VizWiz-Captions Dataset

There are many publicly available image captioning datasets like the popular Microsoft's COCO and Flickr datasets. The images are scraped from the internet and captioned in contrived settings using services like Amazon Mechanical Turk. These datasets are used to design modern algorithms to automatically caption these images. However, such datasets don't represent a real use case like the visually impaired and hence these advanced algorithms don't tend to generalize well with images taken by real people especially like the visually impaired. Hence the researchers at University of Texas at Austin curated a new dataset by sourcing images taken by visually impaired in their natural surroundings for the past 10 years. They used Amazon's Mechanical Turk to caption the images with specific instructions focused on the user group which resulted in detailed captions for the images as compared to ones from COCO or Flickr. They called this the VizWiz-Captions dataset[11].

The dataset consists of around 40,000 images with each image consisting of 1 to 5

captions. These are split into train, val and test splits consisting of 23 431, 7750, 8000 images respectively with a combined caption set of around 196,000 captions. Fig.2 shows some sample images with captions from the train split. As one can see, the images don't seem like the usual ones from COCO or Flickr as the images are not in focus, taken at various angles and lighting which are expected as they are sourced from an atypical source. The captions also are more descriptive than those of COCO or Flickr. The captions consist of more nouns, verbs and adjectives to describe in detail, catering to the

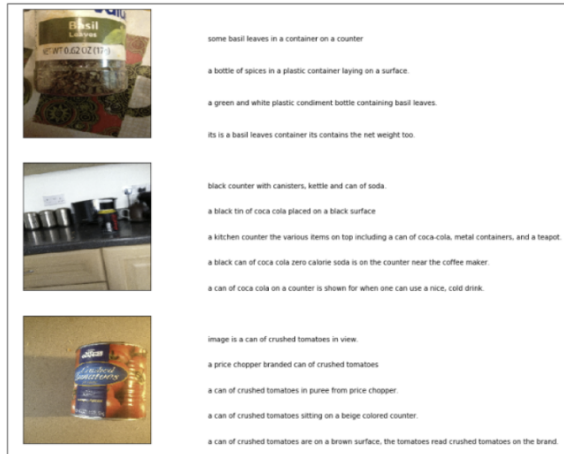


Fig 2: Sample images and captions from VizWiz-Captions train split

needs of the visually impaired like brand names and attributes using verbs and adjectives. An average caption length is about 13 words as compared to 11 words of COCO. It is also noted that about 63% of images contain text and captions for such images have more nouns fewer adjectives than those that do not contain text. The captions also contain metadata to indicate pre-canned(text about insufficient quality images) or rejected captions(text that deemed span)[Examples can be found in Appendix].

With these characteristics, the image captioning model on this dataset should be advanced enough to detect objects and their attributes(colors, up/down, right/left, in/out etc.)

in their spatial regions and identify relationships(actions) amongst them. The captions generated should be novel enough to provide descriptions aimed at the need of the visually impaired. The curators found that some of the state-of-the-art models like Top-down[13], Scene Graph Auto Encoder[14] trained on MS COCO-Captions did not generalize well on the VizWiz-Captions. Hence they created a VizWiz-Captions Challenge[15] to design a model which will caption the VizWiz images with novel captions. The challenge concluded in June 2020 evaluation was done on CIDEr-D[16] metric on test split.

Even though the challenge was over before we started work on our system, we use the dataset as it fits into our design for the use case. We explore and experiment on popular state-of-the-art image captioning algorithms and techniques to train a model that would evaluate well on the test split but also a model that will be viable to run on the Jetson efficiently to caption images in real time.

### 3. Image Captioning Models

The task of image captioning is a congruence of the art of Computer Vision and Machine Translation and there has been a plethora of research in this area[17,18] with different architectures. The most popular ones follow the **Encoder-Decoder** model[19]. This model was made popular by the Neural Image Caption(NIC) model from Show and Tell by Vinyals et al.[20]. Such a model has the encoder with a pre-trained Deep Convolutional Neural Network(CNN) model like VGG16, Inception V3 etc. pre-trained on Imagenet dataset for Image Classification. It takes images as input and converts them into fixed length tensors representing the image features like objects, attributes and salient regions of the objects.

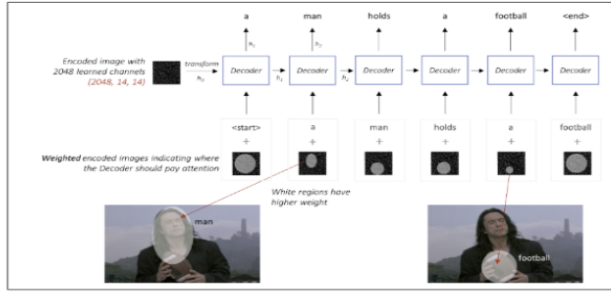


Fig 3a: Attention Mechanism Source: <https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning>

However the last dense output layer which classifies the objects is removed and instead the output from the last hidden layer is fed as input to the Decoder. The Decoder is usually a Recurrent Neural Network(RNN) like Long Short-term Memory(LSTM) or Gated Recurrent Unit(GRU), which has to look at the encoded image and generate caption word by word into a sequence of the final caption. Each predicted word at every time step is used to predict the next word along with the previous state of the RNN.

### 3.1 Attention Mechanism

Attention mechanisms in encoder-decoder models came into forefront made popular by **Show, Attend and Tell**[21] by Xu et al. for visual captioning that exhibited state-of-the-art results on MS COCO and Flickr datasets. With this mechanism[Fig. 3a], the Decoder will be able to look at different parts of the image at different time steps while generating the caption sequence. This is done by using a weighted average across the pixels in the image keeping weights of important pixels greater. Then this weighted representation of the image is combined with the previously generated word at each time step to generate the

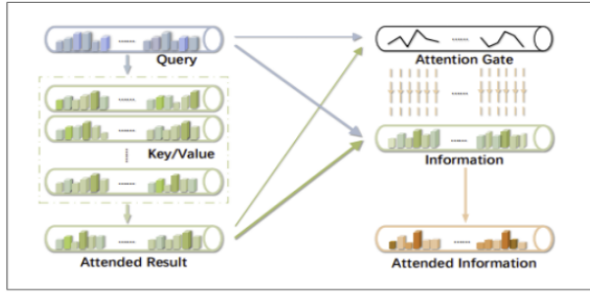


Fig 3b: Attention-on-Attention Mechanism Source: <https://arxiv.org/pdf/1908.09954.pdf>

next word. However the drawback with this model is that Decoder has little idea as to how this attended vector is related to its attention query which in some cases can produce erroneous results.

### 3.2 Attention-on-Attention(AoA) Mechanism

To counter the drawback of the conventional attention mechanism explained earlier, **Attention-on-Attention(AoA)** mechanism was introduced by Huang et al.[22] to determine the relevance of attention results.

AoA generates an *information vector* and an *attention gate*. The information vector is derived from the current context which is the query or the encoded image features and the attention result via a linear transformation and stores this information obtained from the attention result along with the information from the current context. The attention gate is also derived from the attention query and the attention result via another linear transformation with sigmoid activation followed and the value of each channel in the information vector. Then AoA adds another attention by applying the

attention gate to the information vector using element-wise multiplication and finally obtains the *attended information* which represents the expected useful knowledge[Fig. 3b].

AoA is applied to both encoder and decoder in image captional model. AoA on decoder helps to filter out irrelevant attention results to keep only the useful ones and on the Encoder it helps to model the relationships amongst objects in the image better.

The VizWiz-Captions challenge organizers have provided a baseline model based on the Attention-on-Attention mechanism described above called **Attention-on-Attention Net(AoA Net)**. One of our objectives is to run experiments in training this model to beat the performance of the baseline model. Also to evaluate whether such a trained model will be viable for use for inference on the Jetson TX2 for our real time voice activated image captioning system.

#### 4. Training Experiments on AoA Net

To extract the image features, a pre-trained **Faster R-CNN** model using ResNet-101 on Imagenet and Visual Genome is used. This model is based on Bottom-up Top-down approach[24] which had once reached state-of-the-art in image captioning and VQA in 2018. The image features are extracted using bottom-up and consist of encoding spatial regions in an image. The spatial regions consist of identifying instances of objects belonging to certain classes along with their attributes and localizing them with bounding boxes [Fig. 4a].

This model[24] was built using the Caffe framework on Ubuntu 14.4. Since the codebase for the model is constrained to a specific older version of Caffe, it was not possible to compile the libraries because of dependencies with newer Python libraries in the timeframe we had. Hence, we explored other models which would also produce such image feature vectors consisting of objects, attributes and bounding boxes. Most of the models only produce classes and bounding boxes but not the attributes. Having the attributes in the feature vector will help the AoA Net generate more descriptive or novel captions. However we attempted to extract image feature vectors using the current state-of-the-art system of Facebook AI Research Lab's Detectron2[25] which also employs bottom-up mechanism and has support for Caffe.



**Fig.4a** Feature extraction using bottom-up[2]. The features are objects(eg. glasses), attributes(eg. black) and bounding boxes.



**Fig.4b** Feature extraction using Detectron2(bottom-up)[3]. The features are objects(eg. Monitor, microphone, mouse) and bounding boxes.



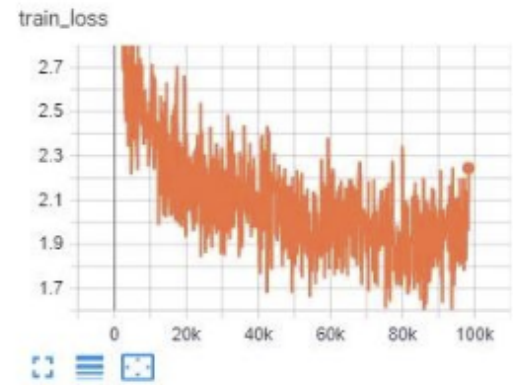
An example of an extracted feature vector for an image from the VizWiz-Captions dataset is shown in Fig. 4b. From the figure we can see that only the objects without any attributes like color/characteristic are present with the bounding boxes unlike the more descriptive features from Fig.4a.

Hence for the training of AoA Net, we proceeded with using the pre-extracted features provided by the baseline model.

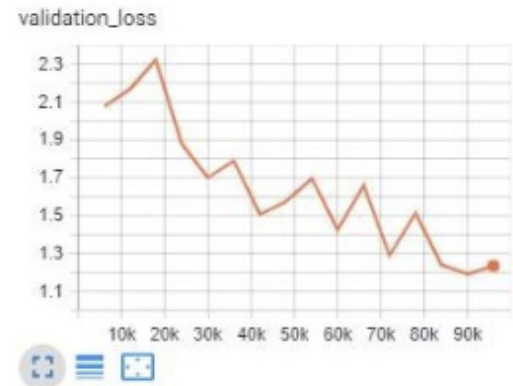
The dimension of the original feature vectors from bottom-up is 2048 but this is projected to a new space of dimension 1024 which is also the hidden size of the 2-layer LSTM used in the decoder. The vocabulary of the captions text in train+val split is trimmed down to 7279 words by removing words that occur less than 5 times. The maximum sequence length in caption vocabulary is 16. These words are encoded into one-hot vectors for the caption generation process in the decoder. The loss function used is cross-entropy loss[26]. The model is first trained on VizWiz-Captions dataset for 25 epochs with a mini-batch of 20. Optimizer used is Adam with learning rate of  $2e-4$  with decay of 0.5 after every 3 epochs and scheduled sampling probability of 0.5 every 5 epochs. After this pre-training, another round of training is done using **Self-Critical Sequence Training(SCST)**[27] for another 40 epochs with batch size of 20 and learning rate of  $2e-5$  and beam size of 2. SCST allows to optimize the performance of the training against CIDEr-D score. The training progress is monitored through Tensorboard[Fig. 5]. Evaluation is done against the test split using beam search with beam size of 3 and batch size of 100 to generate captions against CIDEr-D score. The evaluation results with generated captions for images in test split is submitted to the challenge's evaluation server[28] to get the

results. The evaluation scores for the different trained models in our experiments is shown in Fig. 7. Captions generated for a test split sample is also shown in Fig. 8. Some more experiments were done by increasing/decreasing the number of epochs, changing beam search size to 2 and increasing learning rate decay but none of these resulted in improvement of the model performance.

train\_loss



validation\_loss



**Fig 5: Tensorboard graphs of showing train/val loss of AoA Net scratch model.**

The evaluation results of this AoA Net(Scratch) model trained from only VizWiz captions could not beat the baseline model.

Another experiment was done to fine-tune the model using MS COCO-Captions'14 dataset with the VizWiz-Captions. As per the evaluation results, this didn't perform better than the AoA Net(Scratch) model.

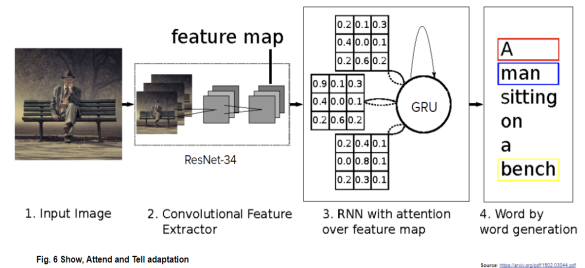
All the training was done on Amazon EC2 p2.xlarge instance using Deep Learning AMI (Ubuntu 18.04) Version 27.0 running on Pytorch 1.4 with CUDA enabled.

**Conclusion:** This model though performed fairly well to caption the test split of VizWiz-Captions as it scored very close to the baseline model score, cannot technically be used for inference in the Jetson as:

- The image feature vectors could not be extracted completely. Only the objects and their bounding boxes could be extracted without the attributes using the Detectron2(bottom-up) model. Limitations in compiling the Caffe libraries hampered in generating the complete set of image vectors needed for the AoA Net trained model for any images in test split or any other external images.
- The AoA Net model is also quite dense as it uses ResNet-101 in Faster R-CNN image feature extraction model and also 2 layer LSTM with 1024 hidden nodes. Such a dense model would be a problem to fit into Jetson's limited memory for efficient inference performance.

Hence, the next experiment focuses on a more technically and practically viable model based on Show, Attend and Tell which has potential to perform efficiently on the Jetson for inference on our voice activated image captioning system.

## 5. Training Experiments with Show, Attend and Tell Model



The final model we used for inference was based on the 'Show attend and Tell' paper. In this model, a vector representing the input image is used as the input.. The vector is generated by an encoder, which is just a Resnet34 pre-trained CNN model. This vector is taken through an RNN, which gives out a vector of activations. The RNN is trained in such a way that this activation represents a sentence that captures the meaning and structure of the image. Since the model has deeper networks, a GRU is used in place of RNN to make training efficient, along with standard regularization techniques. BLEU metric was used to Measure how good the translation is, for the target caption.

### 5.1 Training techniques

For training, a technique called teacher forcing was used. Attention and Beam search were also used to obtain good captions out of images. Teacher forcing is a strategy for training recurrent neural networks that use the model output from a prior time step as an input. For a predicted caption, the RNN predicts each word of the caption in sequence. It uses the words up to some point to predict the next word. If the initial prediction goes off-track, all the subsequent words will also become off track. One way to counter this is by feeding the actual target value at the start of the training and progressively reducing the actual value and

increasing predicted value as the training progresses. The Beam search algorithm selects multiple alternatives for an input caption based on a conditional probability. The number of alternatives depends on a parameter called beamwidth, in our case, it is set to five.

The following Parameters affected final model architecture

- The training time at the cloud
- Jetson RAM utilization
- Jetson GPU activity at inference time

For training, various model architectures were considered. Simpler models were used to make the model fit in Jetson's RAM. Most of the RAM is consumed by the encoder, in this case a CNN model. Complex CNN models such as Resnet101 were considered initially for encoder and final implementation used the Resnet34 Pytorch model. "fast.ai" framework was used for model development - this framework provided high-level components to build the training and inference machinery.

Since each image in the dataset had multiple captions, we randomly selected a caption per image per epoch. This resulted in low BLEU scores while training but yielded very less overfitting.

### Training time

The initial training time was 30 minutes. The following strategies were implemented to reduce training time :

- Converted image files into pickle files for faster loading
- Moved the frequently accessed data to local disk (The pickle files). Used more workers in the data loader. The parallel workers fetch the data from the disk in advance in a pipelined fashion, giving better throughput

- Conducted training over an NFS mount and only frequently accessed data was transferred to the GPU machine. This helped in efficient data and resource management.

These techniques helped us in bringing training time to 6 minutes per epoch. We were able to do more epochs and get a better score.

### 5.2 Usage of one cycle policy

We trained using batches of epochs and after each batch, 'One cycle policy' was run. The optimal learning rate for the next batch was computed and the next batch of epochs were run. This resulted in faster convergence. The one cycle policy does a mock training by going over a large range of learning rates, then plot them against the losses. The trick is to pick a learning rate at which the loss still goes down or is the steepest.

### 5.3 Setting up the model for inference

The inference was attempted on GPU and CPU. It has been observed that predictions run on the CPU were of a lower quality than the GPU. The model structure and parameters were saved. The model is saved as a reusable library. The parameters were saved using Pytorch native format. In the EDGE device, the model was re-constructed and trained parameters were loaded back on the model to do inferencing.

- For testing the model, it was deployed as a flask service that accepts jpeg images. Images were sent to the model and the model returned corresponding captions.
- At the rate of 5 fps, continuous frame per frame inferencing was attempted on a streaming video. The captions were



superimposed on top of each frame of video, just like YOLO demo.

## 5.4 Preparing for Inference

Once the model has been trained in the cloud, need to be prepared for inference at the edge. In the edge, the model needs to be reconstructed and trained parameters need to be loaded. This process is mimicked in the Cloud to weed out problems that may arise. The model is then packaged as a reusable library. The associated environment around the model is then containerised into a docker container and then moved to the EDGE device for further development.

## 5.5 Benchmarking and results

In the Fig. 8, listed are the images and generated captions for various methods we considered for the project. A wide variety of captions are generated by different methods for the same image. We used the Vizwiz image captioning challenge for benchmarking our results. This is shown in Fig. 7. The model we used for production scored much lower in the competition, but this turned out to be the most viable model to be implemented on the edge.

	Aoa(Scratch): a bottle of lotion is on top of a table Aoa(FT): a bottle of lotion is on top of a table SAT: a white bottle with a white label on it		Aoa(Scratch): a hand holding a white bottle with a white cap Aoa(FT): a white bottle is on top of a wooden table SAT: a person is holding a bottle of lotion in their hand
	Aoa(Scratch): a computer screen with white text on a black background Aoa(FT): a green screen with the words signal on it SAT: a computer screen with a black background and white text.		Aoa(Scratch): a jar of something is on top of a table Aoa(FT): a mug with a blue cap on top of a table SAT: the top side of a red and white food container with a red and white label
	Aoa(Scratch): a can of Pepsi sitting on a desk with a desk Aoa(FT): a can of coke is on top of a table SAT: a can of coca cola sitting on a table next to a computer keyboard		Aoa(Scratch): a children's book with colorful and blue and yellow and blue Aoa(FT): a children's book with a colorful butterfly on it SAT: a birthday card with a cartoon character on it

Fig 8: Sample images and captions from VizWiz-Captions test split comparing the 3 trained models

## 6. Edge Inference

The modeling aspect of this socially important use case for image captioning is difficult due to the nature of the quality of images taken by the visually impaired. It is understood that people who are blind cannot verify the quality of the images they take. Regardless of the difficulty of the nature of this problem, research shows that people who are blind prefer to receive descriptions of their image, however inaccurate [11]. This prompted us to think more holistically about the problem at hand – how can we most efficiently convey these captions to the visually impaired? And what levels of integration seem appropriate for this modern world of technology?

### 6.1 Overview

The amount of data and processing power required for training neural networks is massive, but the cost is often outweighed by the accuracy of many of these models. Once trained, a productionalized neural network can achieve 90-98% correct results. The applications for such models often require near real-time predictions and this poses a challenge when inferencing in the cloud. There are several considerations such as cost, latency, scalability and security which make it a more time- and process-intensive solution. The IoT world has

Rank	Team	Bleu-1	Bleu-2	Bleu-3	Bleu-4	ROUGE-L	METEOR	CIDEr	SPICE
1	IBM Research AI	-	-	-	-	-	-	81.13	-
2	SRG-B-VCLab	-	-	-	-	-	-	72.89	-
3	Albums(Boston Univ.)	-	-	-	-	-	-	64.27	-
-	Baseline(AoA-Net)	65.91	47.77	33.68	23.41	46.56	20.00	59.77	15.11
-	Team SSP (AoA-Net Scratch)*	65.70	47.15	32.97	22.80	46.38	19.79	<b>59.56</b>	14.88
-	Team SSP (AoA-Net Fine-tuned)*	65.68	46.94	32.85	<b>22.83</b>	<b>46.46</b>	19.82	58.50	<b>14.91</b>
-	Team SSP (Show, Attend And Tell)	58.58	39.57	26.49	17.71	40.75	16.57	37.18	11.06

\* - Post Competition Scores. Scores for other evaluation metrics not available. Post Competition Ranks not available.  
Fig. 7: Evaluation Results comparing the 3 models against baseline model top ranks of the VizWiz-Captions Challenge.

For other models, because of tool and setup issues, parameter extraction was difficult. Those models were complex and had more number of parameters making them not suitable for inference at EDGE.

provided us with multiple devices which leverages edge computing, a paradigm that has brought computation and data storage closer to the required location for inference. Edge computing addresses the challenges of the cloud but comes with its own complexity in terms of being able to provide enough compute power to run the models [34]. However, companies such as Nvidia, Intel and Google are now designing chips which can execute operations in milliseconds [35]. It is this training coupled with inference which lends strength to the field of deep learning with respect to its ability to produce valuable insights to everyday problems [32].

Inferencing at the edge also plays an important role in furthering the impact of models generated for the VizWiz-Captions dataset. Small IoT devices that can be used in the safety of one's home network with quick response times are an ideal candidate to perform predictions for images taken by the visually impaired. Our next step in the project was to develop a proof of concept to demonstrate this. The edge device used in this project is the NVIDIA Jetson TX2, an open platform that delivers AI computing at the edge. It is a full Linux machine built around the NVIDIA Pascal GPU, one of the company's fastest processing units with the ability to efficiently process images, video and other data required for neural networks. We leveraged the Jetson to not only provide near real-time inference but developed a pipeline that would caption a live image on-demand when an Alexa utterance was verbalized.

## 6.2 Edge Architecture

Figure 9 shows an overview of the edge architecture wherein a user requests Alexa to take a picture on demand which is then

captioned using the previously trained Show and Attend model. We now describe the different parts of the architecture.

### *Alexa Skills Kit (ASK)*

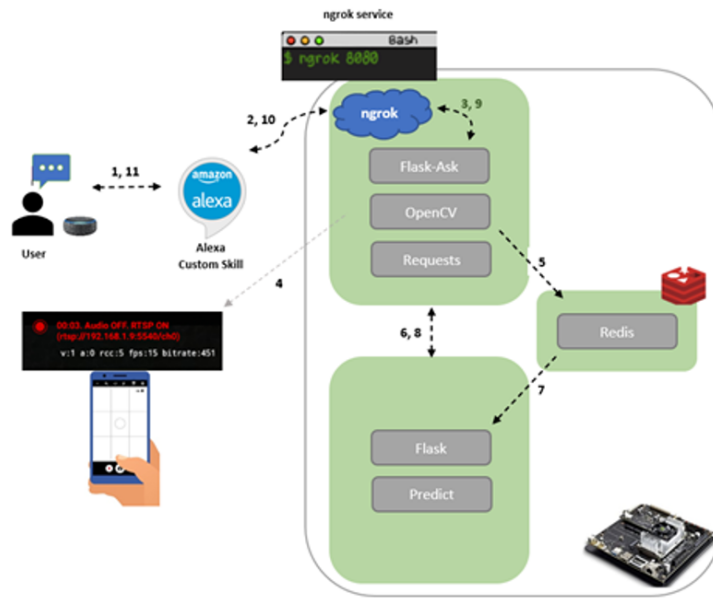
For this project we used the Alexa Skills Kit (ASK) to build a custom skill with a unique invocation “*what do you see*”. The configuration requires that we use a HTTPS endpoint which we provisioned locally using ngrok (discussed below) as well as a JSON file definition outlining the request to be sent to the endpoint.

### *Jetson Containerization*

In the Jetson, on the right we have three containers. The first container utilizes ngrok, Flask-Ask and OpenCV.

- ngrok is a reverse proxy that creates a secure tunnel from a public endpoint to a locally running web service. It essentially exposes the local server to the internet.
- Flask-Ask is a python library for Alexa skill development. It comes with many decorators to map Alexa requests so we can return both statement and question constructs back to the Alexa service.
- OpenCV is a Python library that is used to solve computer vision problems through numpy operations. We are able to use the methods it provides to configure various camera devices (USB/WiFi) to take streaming video and capture images.

The second container brings up a redis store. Redis is a popular choice for machine learning applications as it provides a fast in-memory data store that we can leverage to decrease data access latency, increase throughput and ensure



**Figure 9: Edge architecture. Steps detailed in Section 6.2.**

that the containers remain independent in terms of system design.

The third container is the prediction container holding the model weights. The prediction code is encapsulated with a Flask wrapper to create an endpoint on the local network. This allows it to be called from any application on the home network to caption an image.

### *Process Overview*

We start on the left with a user asking Alexa, “What do you see?” The Alexa device records the audio and sends it to Alexa cloud for processing. Based on the invocation name, the Alexa service passes a JSON file to the ngrok endpoint in the first container. Ngrok forwards the JSON file to the python script that is also running in the first container, but on our local network. This python script uses Flask-Ask to process the request and prompts our camera to take a picture on-demand via OpenCV. The camera we used for this was on our smartphone and we used Real Time Streaming Protocol apps

on Android and Apple to configure that. The image is then encoded and pushed to redis. A request is sent from the first container to the third to process the image. In the third container, the image is retrieved from redis and the model predicts a caption text using the Show, Attend and Tell model. This text is sent back to the first container and then returned via ngrok to the Alexa cloud which sends it to the Echo dot to communicate back to the user.

### *Technical Steps*

The specific steps seen in Figure 9 are outlined below.

1. User invokes Alexa custom skill, Echo dot processes audio and sends it to Alexa service.
2. Alexa Skills Kit sends message to ngrok endpoint
3. ngrok forwards JSON to Flask-Ask (processes skill intents)
4. OpenCV takes a picture on-demand and encodes it as a JPG into memory buffer
5. Buffer saved in base64 encoding in redis

6. Request sent to prediction container via Flask endpoint to make a prediction
7. Image retrieved from redis and decoded, prediction model invoked, image is captioned
8. Caption sent back to ngrok container
9. Caption sent back via Flask-Ask to ngrok
10. ngrok sends caption back to Alexa service
11. Alexa sends caption back to Echo Dot

The full code repository including model weights, Docker files, Python scripts and instructions for bringing up the containers and running the setup end to end are outlined in the accompanied Github repo.

### 6.3 Results

The end to end system was tested on a variety of objects. The average inference time was around 5s. The following YouTube videos capture the model correctly identifying common items that a person might encounter in their day to day activities.

- Laptop, can of soup, coffee mug, teddy bear  
<https://youtu.be/7HJAT0qpQac>
- Kitchen fridge and stove  
<https://youtu.be/3g8MxeDQ4Cg>
- Can of water, vitamin bottle  
<https://youtu.be/IdNFus2YNZQ>

The results from the inference testing look promising and reveal the strength of edge computing in filling a gap where instantaneous object detection can change lives and make significant improvements in the lives of the visually impaired.

## 6. Conclusion and Improvements

- To productionalize an end-to-end model for inference several considerations must be taken into account. First, whether the system should be able to scale to multiple users and if that would imply moving the predictions to a cloud based system. However, this raises a lot of security concerns in terms of image data being passed to the cloud. Although we are able to identify the source of an Alexa request based on the Skill Id (unique to a user and skill), adding security checks for each request might increase the latency for the response to come back to the Alexa device. Security is a general second consideration as well since we would need to move away from using ngrok to provide a HTTPS endpoint and use a more secure method such as nginx with Kubernetes. Doing so might also allow us to integrate with IP cameras outside the home network, thereby improving the range and connectivity of the application.
- AoA Net model's training performance was boosted by the SCST mechanism based on optimizing the CIDEr-D score. There is a proposal of a new approach based on Bayesian-SCST[29] to make the model train better.
- Since around 63% of images contain text, it would be ideal to pre-process the images for the text processing using OCR mechanisms like scene text detection mechanism[30].
- Image Augmentation techniques were not applied for our trained models as the AoA Net baseline model failed to perform better with augmentation like blurring, rotation. However, a refined approach to applying augmentation is do

it on a case by case basis on the images. For example, rotation can be applied but the captions generated for the rotated images need to be ranked based on whether they provide valid descriptions of the image. Blurring can also be applied but to in-focus images to improve detection of objects and attributes better.

- Use a different pre-trained model for image feature extraction like models trained on Instagram photos which are also taken in real human environment settings.
- Use a transformer in the encoder-decoder model which has shown to exhibit state-of-the-art in image captioning on MS COCO dataset[31].



## 7. Acknowledgements

This project was developed as part of the final project requirements of the course **W251: Deep Learning in the Cloud and at the Edge, Summer 2020** under the guidance of *Brad DesAulniers* and *Darragh Hanley*.

## 8. References

1. How does automatic alt text work on Facebook:  
<https://www.facebook.com/help/216219865403298>
2. Add alternative text to a shape, picture, chart, SmartArt graphic, or other object:  
<https://support.microsoft.com/en-us/office/add-alternative-text-to-a-shape-picture-chart-smartart-graphic-or-other-object-44989b2a-903c-4d9a-b742-6a75b451c669>
3. Aira: <https://aira.io/>
4. C. L. Bennett, M. E. Mott, E. Cutrell, and M. R. Morris. How Teens with Visual Impairments Take, Edit, and Share Photos on Social Media. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, page 76. ACM, 2018.
5. J. P. Bigham, C. Jayant, H. Ji, G. Little, A. Miller, R. C. Miller, R. Miller, A. Tatarowicz, B. White, and S. White. VizWiz: Nearly real-time answers to visual questions. In Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology, pages 333–342. ACM, 2010.
6. BeSpecular. <https://www.bespecular.com>.
7. Nearly 70% of US Smart Speaker owners use Amazon Echo devices:  
<https://techcrunch.com/2020/02/10/nearly-70-of-u-s-smart-speaker-owners-use-amazon-echo-devices/>
8. Alexa For Blind Seniors And Adults Who Are Visually Impaired:  
<https://seniorsafetyadvice.com/alexa-for-blind-seniors/>
9. NVIDIA Jetson TX2 Module: <https://developer.nvidia.com/embedded/jetson-tx2>
10. Using GPUs for training models in the cloud:  
<https://cloud.google.com/ai-platform/training/docs/using-gpus>
11. D. Gurari et al. Captioning Images Taken by the Blind July 2020:  
<https://arxiv.org/abs/2002.08565v2>
12. Xu et al. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention Feb 2015: <https://arxiv.org/abs/1502.03044>
13. Anderson et al. Bottom-up and Top-Down Attention for Image Captioning and Visual Question Answering March 2018: <https://arxiv.org/abs/1707.07998>
14. Yang et al. Auto-encoding scene graphs for image captioning Dec 2018:  
<https://arxiv.org/abs/1812.02378>
15. VizWiz-Captions Challenge: <https://vizwiz.org/tasks-and-datasets/image-captioning/>
16. Vedantam et al. CIDEr: Consensus-based Image Description Evaluation:  
<https://arxiv.org/abs/1411.5726>
17. Staniute et al. A Systematic Literature Review on Image Captioning March 2019:  
<https://www.mdpi.com/2076-3417/9/10/2024>

18. MTank's Multi-Modal Methods: Recent Intersections between Computer Vision and Natural Language Processing: <https://www.themtank.org/multi-modal-methods>
19. Kshirsagar Automatic Image Captioning with CNN & RNN Jan 2020: <https://towardsdatascience.com/automatic-image-captioning-with-cnn-rnn-aac3cd442d83>
20. Vinyals et al. Show and Tell: A Neural Image Caption Generator April 2015: <https://arxiv.org/abs/1411.4555>
21. Xu et al. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention Feb 2015: <https://arxiv.org/abs/1502.03044>
22. Huang et al. Attention on Attention for Image Captioning August 2019: <https://arxiv.org/abs/1908.06954>
23. Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposed Networks Jan. 2016: <https://arxiv.org/abs/1506.01497>
24. Anderson et al. Bottom-up and Top-Down Attention for Image Captioning and Visual Question Answering March 2018: <https://arxiv.org/abs/1707.07998>
25. Facebook AI Research's Detectron2: <https://github.com/facebookresearch/detectron2>
26. J Brownlee A Gentle Introduction to Cross-Entropy for Machine Learning: <https://machinelearningmastery.com/cross-entropy-for-machine-learning/>
27. Rennie et al. Self-Critical Sequence Training for Image Captioning(SCST) Nov. 2017: <https://arxiv.org/abs/1612.00563>
28. Eval AI Server for VizWiz-Captions Challenge Evaluation: <https://evalai.cloudev.org/web/challenges/challenge-page/525/overview>
29. Bujimalla et al. B-SCST: Bayesian Self-Critical Sequence Training for Image Captioning June 2020: <https://arxiv.org/abs/2004.02435>
30. Long et al. Scene Text Detection and Recognition: The Deep Learning Era Sept. 2019: <https://arxiv.org/abs/1811.04256>
31. He et al. Image Captioning through Image Transformer April 2020: <https://arxiv.org/abs/2004.14231>
32. Giant Leaps in Performance and Efficiency for AI Services, from the Data Center to the Network's Edge: <https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/tesla-product-literature/t4-inference-print-update-inference-tech-overview-final.pdf>
33. GPU Inference Analysis: A Perspective At The Rugged Intelligent Edge: <https://premioinc.com/blogs/blog/inference-analysis-for-rugged-edge-computing>
34. Sarmah, Dilip. Why Machine Learning at the Edge: Oct. 2019: <https://www.predictiveanalyticsworld.com/machinelearningtimes/why-machine-learning-at-the-edge/10669/>
35. Maquaire, Nicholas. Inference on the edge. May 2019: <https://towardsdatascience.com/inference-on-the-edge-21234ea7633>

## 10. Appendix




	<p>some basil leaves in a container on a counter</p> <p>a bottle of spices in a plastic container laying on a surface.</p> <p>a green and white plastic condiment bottle containing basil leaves.</p> <p>its is a basil leaves container its contains the net weight too.</p>
	<p>black counter with canisters, kettle and can of soda.</p> <p>a black tin of coca cola placed on a black surface</p> <p>a kitchen counter the various items on top including a can of coca-cola, metal containers, and a teapot.</p> <p>a black can of coca cola zero calorie soda is on the counter near the coffee maker.</p> <p>a can of coca cola on a counter is shown for when one can use a nice, cold drink.</p>
	<p>image is a can of crushed tomatoes in view.</p> <p>a price chopper branded can of crushed tomatoes</p> <p>a can of crushed tomatoes in puree from price chopper.</p> <p>a can of crushed tomatoes sitting on a beige colored counter.</p> <p>a can of crushed tomatoes are on a brown surface, the tomatoes read crushed tomatoes on the brand.</p>

Fig 2: Sample images and captions from VizWiz-Captions train split



quality issues are too severe to recognize visual content.



quality issues are too severe to recognize visual content.



quality issues are too severe to recognize visual content.

quality issues are too severe to recognize visual content.

quality issues are too severe to recognize visual content.

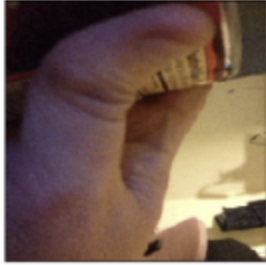


quality issues are too severe to recognize visual content.

Sample of Pre-canned captions from VizWiz train split.



a wonderful view of the fog windows in the room is very thick



imagine how you would describe this image on the phone to a friend



beautiful view from behind the walls hidden under dark mist



beautiful view from behind the walls hidden under dark mist

Sample of Rejected captions from VizWiz train split.



	<p>AoA(Scratch): a bottle of lotion is on top of a table</p> <p>AoA(FT): a bottle of lotion is on top of a table</p> <p>SAT: a white bottle with a white label on it</p>		<p>AoA(Scratch): a hand holding a white bottle with a white cap</p> <p>AoA(FT): a white bottle is on top of a wooden table</p> <p>SAT: a person is holding a bottle of lotion in their hand</p>
	<p>AoA(Scratch): a computer screen with white text on a black background</p> <p>AoA(FT): a green screen with the words signal on it</p> <p>SAT: a computer screen with a black background and white text .</p>		<p>AoA(Scratch): a jar of something is on top of a table</p> <p>AoA(FT): a mug with a blue cap on top of a table</p> <p>SAT: the top side of a red and white food container with a red and white label</p>
	<p>AoA(Scratch): a can of pepsi sitting on a desk with a desk</p> <p>AoA(FT): a can of coke is on top of a table</p> <p>SAT: a can of coca cola sitting on a table next to a computer keyboard .</p>		<p>AoA(Scratch): a <u>children's</u> book with colorful and blue and yellow and blue</p> <p>AoA(FT): a <u>children's</u> book with a colorful butterfly on it</p> <p>SAT: a birthday card with a cartoon character on it</p>

**Fig 8:** Sample images and captions from VizWiz-Captions test split comparing the 3 trained models