Tiki: A Prototype Virtual Assistant for VQA

Brent Biseda, Pri Nonis, Kevin Stone, Vinicio De Sola (UC Berkeley, MIDS)

https://github.com/brentbiseda/tiki

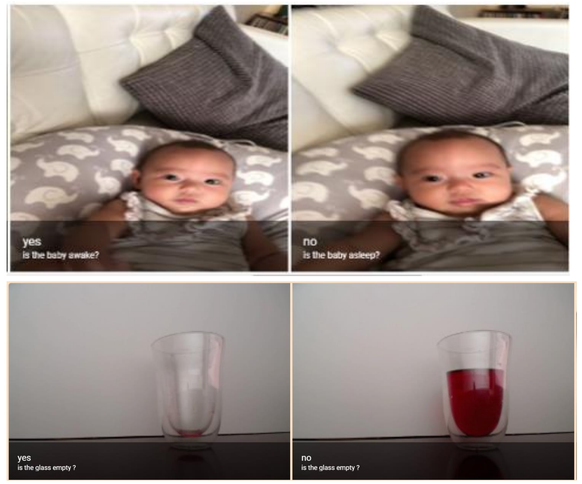
Background

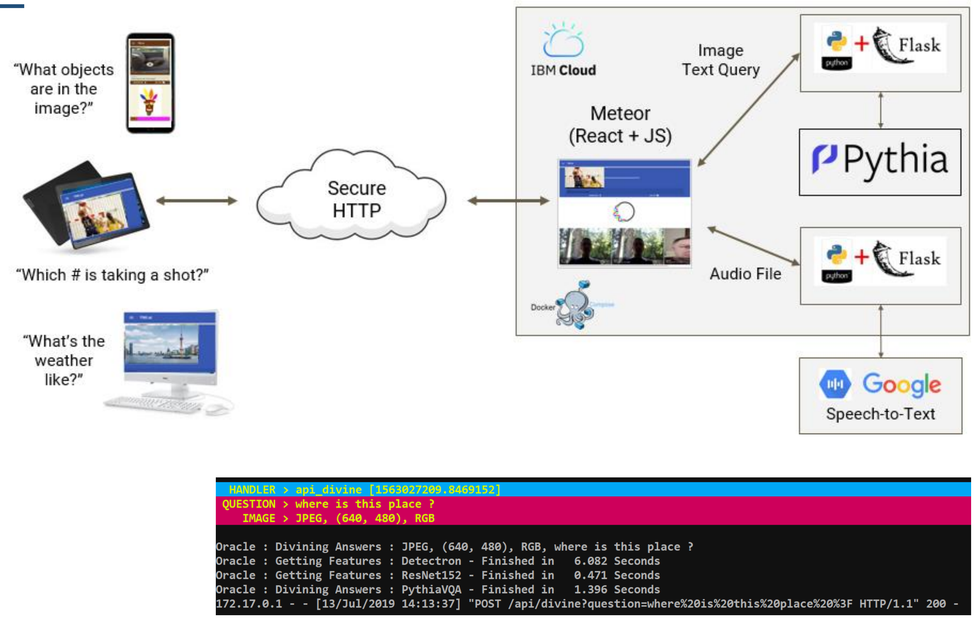
The advancements in Deep Learning have changed many areas of research and applications and span the range from Image Recognition to Natural Language Processing. One of the most recent frameworks released by Facebook AI Research received state of the art results on Visual Question and Answer (VQA) tasks. This model, called Pythia, is part of our model to create a virtual assistant that can answer basic questions when asked.

Visual question answering (VQA) is a new field that combines both computer vision and NLP to provide answers to simple questions using common human syntax. This project is an implementation of a state-of-the-art VQA model (Pythia) in a web app. Over time, this type of service will become more common throughout all smart devices such as Alexa, Siri, and others. Tiki demonstrates the potential for this rollout to be imminent. Here we demonstrate the capacity to use a voice query to provide answers for any image on a remote camera.

Virtual Assistant Overview

- Tiki is a virtual assistant prototype
- Performs inference in cloud on CUDA device
- 1.5 second latency for inference on GPU (V100)
- 6 second latency for inference on CPU (V100)






```
HANDLER > api_divine [1563027209.8469152]
QUESTION > where is this place ?
   IMAGE > JPEG, (640, 480), RGB

Oracle : Divining Answers : JPEG, (640, 480), RGB, where is this place ?
Oracle : Getting Features : Detectron - Finished in   6.082 Seconds
Oracle : Getting Features : ResNet152 - Finished in   0.471 Seconds
Oracle : Divining Answers : PythiaVQA - Finished in   1.396 Seconds
172.17.0.1 - - [13/Jul/2019 14:13:37] "POST /api/divine?question=where%20is%20this%20place%20%3F HTTP/1.1" 200 -
```

Application Architecture

This project demonstrates an end-to-end implementation of VQA using a mobile phone, or any device capable of rending HTML 5 in browser.

Front End Design:

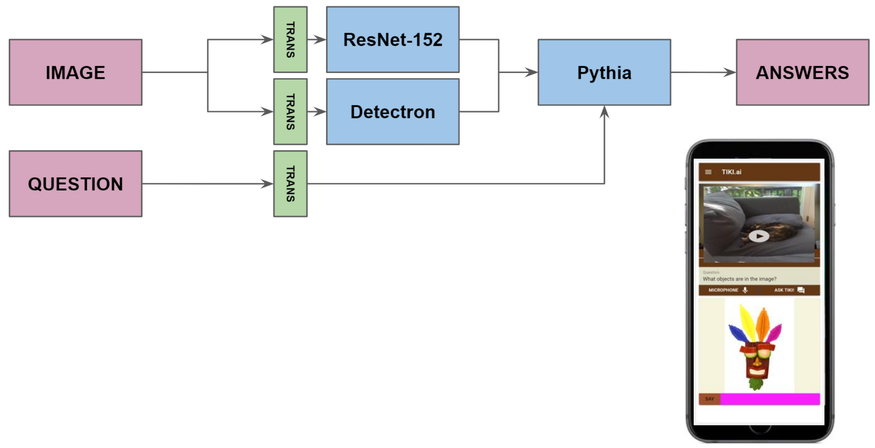
- Meteor – Full-stack JavaScript App that assembles all the pieces needed to build either websites or mobile apps. It automatically manages the data flow between cloud and client applications, client UI and rendering, regardless of the framework being used.
- React – React is a declarative, efficient, and flexibile JavaScript library for building user interfaces by using a modular framework of components.
- Material Design – Open-source design system developed by Google.

Back End Design:

- Flask – Python based lightweight web application framework. This was used to process the image requests sent for VQA inference.
- Docker-compose – Tool developed by Docker to run several containerized pieces simultaneously.
- Pythia – Model designed by Facebook AI research for VQA. This model was containerized and hosted via a flask API.
- Google Speech-Text: Google Cloud based API based on an RNN-T transducer that can parse real-time streaming or prerecorded audio.

Pythia Model Details

- Built on open-source PyTorch framework
- Used object detector to extract image features with bottom-up attention.
- ResNet-101 for backbone network.
- Uses Visual Genome, knowledge base to connect structured image concept to language.
- The question text is then used to compute the top-down attention
- Uses GloVe (Global Vectors) word embeddings -> GRU network and a question attention module to extract text features
- Reached 70.34% on VQA 2.0 with an ensemble of 30 models.



Application and Future Work

- Integration with smart home devices
  - Alexa
  - Siri
  - Nest Cameras
  - Ring
- Industry-specific training:
  - Healthcare triage
  - Home surveillance

As the initial Pythia model was trained using general VQA datasets, the model would be refined by training on domain specific datasets.

References

1. Google AI. An all-neural on-device speech recognizer, Mar 2019. URL https://ai.googleblog.com/2019/03/an-all-neural-on-device-speech.html
2. Docker. Overview of docker compose, Aug 2019. URL https://docs.docker.com/compose/
3. Google. Getting started, 2018. URL https://material.io/collections/getting-started/#01
4. Meteor. Meteor, 2018. URL https://www.meteor.com/meteor-faq
5. Pallets. Flask, 2010. URL https://palletsprojects.com/p/flask/
6. React. Tutorial: Intro to react, 2019. URL https://reactjs.org/tutorial/tutorial.html
7. Towards VQA Models That Can Read. Singh, Amanpreet and Natarajan, Vivek and Shah, Meet and Jiang, Yu and Chen, Xinlei and Batra, Dhruv and Parikh, Devi and Rohrbach, Marcus. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019. [https://github.com/facebookresearch/pythia]