CS 655 Computer Networks
Mini-Project Image Recognition Application

Team Member
Zifeng Chen: zf990318@bu.edu
Yiying Hu: yiyinghu@bu.edu
Github link: https://github.com/zf990318/CS655-Mini-Project/tree/master

1. **Introduction/ Problem Statement:**
   In this project we have implemented an image recognition application with pretrained vgg16 neural network, and deployed it on the GENI node. The system could perform image recognition classification when the user uploads the chosen image, the web interface will send it to the backend node, after classification is done by the recognition system on backend node, it will send classification result, confidence level, throughput and round trip time to the web interface and display on it.Through this project we have learned how to develop front- end application and deploy them on different network load, and also learned the how different size images and different delay affect the round trip time and throughput

2. **Experimental Methodology**

   2.1 Web Interface:
   The web interface has deployed on node 192.41.233.48, it basically provide user a entry to submit the image and get the result.When user initiates a get request to the web interface, a page for upload the image will show up for user, and web interface will send the submitted image to backend server, and display the result after backend send back the result to web interface

   ## 2.2 Recognition System:
   For the recognition system we have implemented a pretrained Resnet 50 neural network and it deployed at 192.41.233.48:5000, there are 1000 optional outputs and we chose the top-1 prediction and its accuracy to display on our web interface.

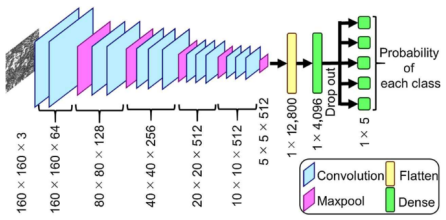


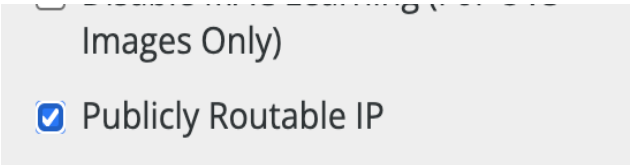
Fig 1: ResNet50 Model

   The input image will fed forward to model, once backend server received it will start predict the category of image, then send the result back to the web interface and display it

3. Result

3.1.1 Set up GENI

We only need to nodes for this project. One of them is for HTML file. We use Nginx to start a html server. Another one is server which will provide a machine learning service. The Rspec files is ------. When I started this project, I forgot to choose Publicly Routable Ip for the nodes which resulted in I can't visit the nodes even the service was started successfully.



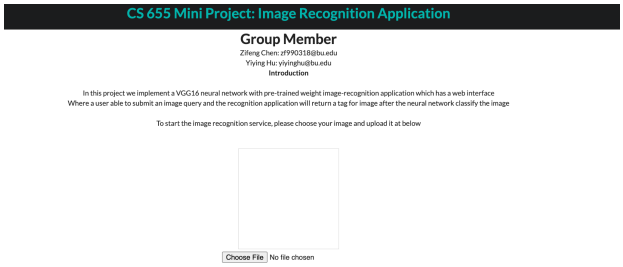
3.1.2 Set up HTML-Server

The html server is actually the client for this project. We chose tp deploy the html on Nginx. SSH to HTML-Server node

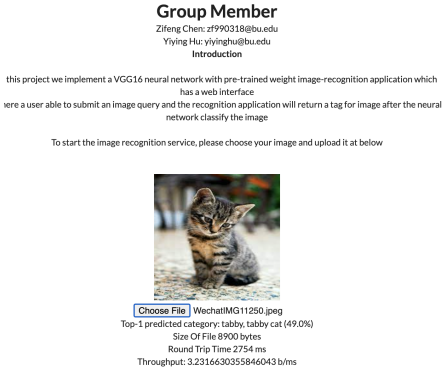
Step 1: Install Nginx
sudo apt update
sudo apt install nginx
Step 2: Adjust the firewall to allow the Nginx service to pass through
sudo ufw app list
sudo ufw allow 'Nginx HTTP'
sudo ufw allow 'Nginx HTTPS'
Step 3: Make Sure the Nginx is running
systemctl status nginx
Nginx is running succesfully if you see
● nginx.service - A high performance web server and a reverse proxy server
Type in the ip of your node in browser, ip of my http-server is 192.41.233.48Nginx works succesfully if you see this page

Step 4: Edit **/etc/nginx/sites-available**
root /users/yiyinghu/HTML/templates;
index index.html
Step 5: Restart Nginx
sudo nginx -s reload
Now, when I type in the ip of html server, I am able to see my own page
3.1.3 Set up ML-Server
As we need do some prediction based on machine learning. So we use Python to do the back-end.
Step1: Install Anaconda3
wget https://repo.anaconda.com/archive/Anaconda3-2021.05-Linux-x86_64.sh
Step2: Set up Environment
Edit /.bashrc
export PATH=/users/yiyinghu/anaconda3/bin:$PATH
Create virtual environment and activate

conda create -n CS655 python=3.7
conda activate CS655
Download packages
Run server
python3 ML-Server.py
3.1.4 Process of project
When the client and server is running successfully, type in the ip of client in local chrome. Then you will see



Choose the file and upload the file. Then the result will display



3.2 Analysis
3.2.1 Different size of image
We test 5 images with different size

	Size(bytes)	RTT(ms)	ThroughOutput(bytes/s)
1	141,794	3209	44
2	8900	2702	3
3	7784	2761	2.8
4	3739763	4761	785
5	464761	3317	140

We can see that when image is big enough, the thoughtoutput is bigger even though rtt is bigger too. Otherwise, when image is smaller than other one, it may takes more than for the smaller one.
3.2.2 Different delay on link
we use the command below to set a delay on our node
sudo tc qdisc add dev eth0 root netem delay 100ms

	Delay(ms)	Size(bytes)	RTT(ms)	ThroughtOutput(bytes/s)
1	200	8900	3570	2.5

2	400	8900	7790	1.14
3	600	8900	8928	0.99
4	200	141794	6413	22
5	400	141794	6818	20
6	600	141794	10105	14

4. Conclusion

After test on different size of images and different time delay on the same images. We come to the conclusion that the thrououtput and rtt is greater when images is bigger. But that also means that our network works well with big images. And when we set delay to our networks, we find that the time delay have much more influence on the processing of small images.

5. Division of Labor

Zifeng Chen: part of front-end, back-end server, machine learning model

Yiying Hu: part of front-end, deployment of server and client, geni design and test