Title : Cloud-based Distributed Video Processing for Object Detection

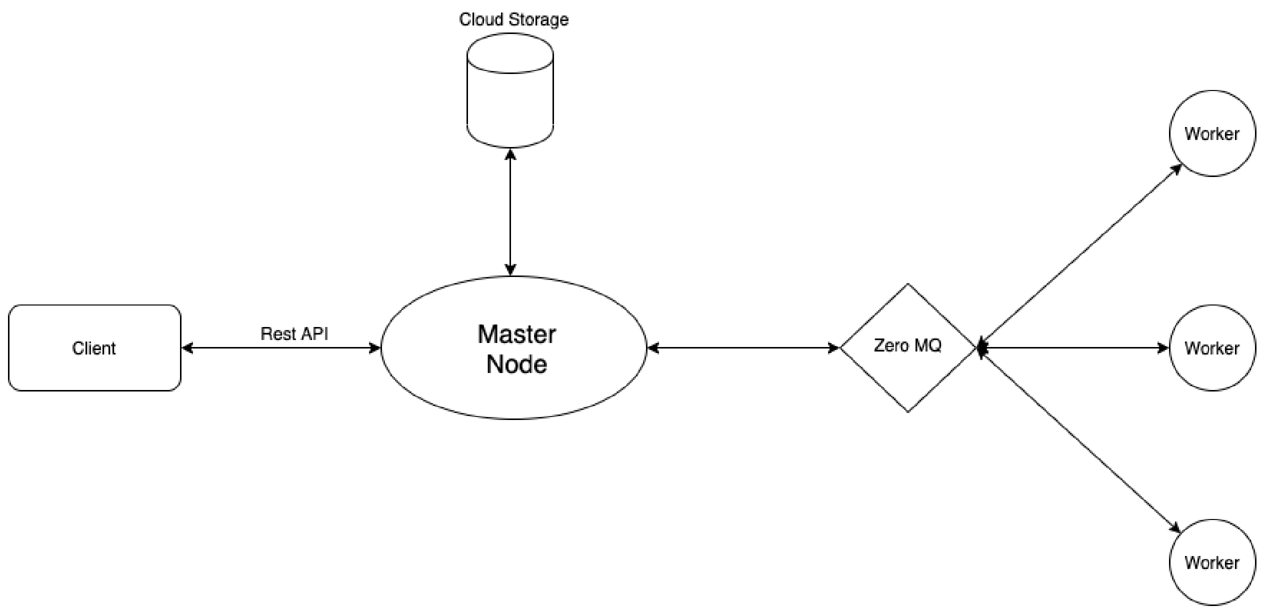
List of Project Participants : Nischal Paramashivaiah , Sitesh Ranjan

Link to YouTube Video :
https://www.youtube.com/watch?v=VoQkvImJO_E&ab_channel=Nischalhp

Project Goals: Through this Project we aimed to create a distributed processing of video frames for object detection using a pre-trained DNN. By inputting a video into the system, we will essentially get back a similar video with different objects highlighted and also their count in a particular frame. These objects could be cars, people or any other object supported by the pre-trained DNN. This will happen by splitting up the video into individual frames and processing each of those individual frames. Once the processing is done on the frames, they are merged back in a way that the original frame order is maintained. Thus, we get the required video with all the objects and their number highlighted.

Software Components : REST API Interface , Cloud Object Storage , VM Instance , ZeroMQ

Architecture Diagram :



Description of the Project and also the Interaction between different Components:

1) The client node sends the request to process a video to the master node (All the nodes here refers to VM's). The request consists of an URL which points to a cloud storage bucket which holds the video. This request is received by the Master node which splits up the video into individual frames. The individual frames are sent to the worker nodes for processing through ZeroMQ which has a special wrapper to handle the images specifically. Until the worker node finishes processing and sends an acknowledgement back, ZeroMQ will not send a new frame for processing to that particular worker node. ZeroMQ sends out the frames in a batch of 100 to each VM. Thus, ZeroMQ helps handle processing of multiple frames simultaneously.
2) Each of the worker nodes and the master node will be a separate VM instance. They will be used to identify the objects in each of the video frames using a pre-trained Neural Network model. Each of the worker nodes will process the assigned video frame and return the processed frame back to the Master node.
3) The Master node will then serialize the frames according to their frame number and reorder it to obtain back a similar video to the input video. This has to be done so that we get back the video in the same order as the original one. Then, this processed video is stored in a cloud storage bucket. The URL of the new video which has objects highlighted in it, will be sent back to the client once the processed video is successfully stored in the cloud bucket.

Description of each Component:

1) <u>ZeroMQ</u> :

Purpose : In our project ZeroMQ is used for implementing the message queue. Through the class assignment we got a taste of a broker system like RabbitMQ. Hence, we wanted to explore the world of brokerless systems and also chose this for some specific reasons listed below.

Advantages:

a) Unlike other implementations of message queue such as RabbitMQ and others; ZeroMQ does not need a dedicated message broker. It is brokerless, low cost and easy to implement as well.
b) It is using positive acknowledgement for sending the frames, so it ensures no loss of video frames.
c) In our project, we will have to send images between different virtual machines. This would be very easy with ZeroMQ, as there is a wrapper called ImageZMQ which is built specifically for this purpose.
d) The final advantage is it scales very well for a distributed application such as ours.

Disadvantages:

a) The relationship between different components needs to be managed more when compared to other systems.

2) <u>VM Instance</u> :

Purpose : In this project, we have used several VM instances, few of which acts as worker nodes and one of them acts as a Master node and another one as the client.

Advantages :

a) As we had used VM's before in several assignments before, we were well versed with implementing it and chose it over Kubernetes.
b) It is also easy to debug if anything goes wrong with VM's as we can easily SSH into the VM and check the logs.This is particularly hard with Kubernetes.
c) It is also very easy to deploy a VM in GCP with the required OS. And also very easy to install other software on top of it.

Disadvantages:

a) Since we have used VM's, we have to manually add and remove worker nodes depending on the load. This could be automated if we had used Kubernetes using something like Load Balancing.
b) The second disadvantage is that there is inefficient resource utilization when it comes to using VM's; as we would not need so much memory for running a simple application.

3) <u>Cloud Object Storage</u> :

Purpose: Here Cloud object storage is basically used for storing the processed video and also to retrieve the video initially.

Advantages:

a) Very large video files could easily be uploaded and downloaded from Cloud Object Storage.
b) Since the VM's are located within the same ecosystem of Google Cloud, there is fast access to the video located in the Cloud storage. This is particularly enhanced if they are located in the same Region or Zone.

Disadvantages:

a) If this project idea was provided as a service; it might have been cumbersome to users. As they would prefer to upload the videos directly from their device rather than upload it to Google Cloud and then access it from there.
b) Even though very large video files could be stored in Google Cloud, theoretically there exists a limit. And to get a larger storage limit, will cost us more and make the service more expensive to operate.

4) <u>REST API</u> :

Purpose: We have used REST API to handle the request between the client and the master node. The client sends the request which is basically a URL of the Cloud Object Storage where the video is stored, which is then processed by the master node.

Advantages:

a) It is very easy to code REST API when compared to something like gRPC.
b) The payload of the request is very small, since it is just a single URL name. Also, assuming very few requests are sent at a time, it is almost as fast as gRPC. This

point was proved when we did a comparison study in a previous assignment of this course.

c) There is greater flexibility provided as REST can handle multiple types of calls and handle different file formats very easily; which was required for this project.

Disadvantages:

a) Since REST makes a new TCP connection for every request, this method might be slow when compared to gRPC especially if client and master are physically at a greater distance from each other.

Debug/Testing in the project:

1) We have used the VM logs (that is SSH into the VM and check their processing) to debug any faults with regards to the Master or the worker nodes. By checking these logs we will be able to ascertain whether the video frame is being split up properly or not. We will also be able to see if objects are being detected in a single frame or not.
2) Similar process was used to make sure that ZeroMQ is working properly as well. We can essentially check logs for the ZeroMQ by SSHing into the VM and ascertain whether messages are being queued up properly and also we can check if all the requests are being handled correctly or not by printing out the requests.
3) We included a Verbose flag in the code. Enabling which, all the processes and requests would get printed out; and this would make debugging or testing easier.
4) The final verification step could be obtained by checking the final video which has objects highlighted in it. If the new video matches the original video in terms of frames, then it means that frame splitting and later the frame rejoin process was done correctly by the master node.
5) If the new video obtained also has all the objects highlighted in it, then it means that each of the worker nodes have successfully completed their task of identifying objects in a single frame.
6) The pre-trained neural network model that we want to use is already trained and tested upon. Hence, we can be sure of the output which it generates and need not have to implement a separate training/testing mechanism for the specific model. The only thing that we can play around with is different parameters associated with the neural network model like confidence measure which ultimately impacts how well the objects are identified in the video.

Meeting Project Requirements :

This Project will meet the Project Requirements as we are using 4 different Cloud technologies as specified below:

1) REST API interface - To service the client
2) Storage services (Object storage) - To store the input and output video
3) Message queues (ZeroMQ) - To handle processing of video frames
4) Virtual Machines - Master and worker nodes

Capabilities of the System :

1) The system can handle video of any size and resolution. Although, providing a video of large size will take up a lot of time for processing and to get the final output. (For example: A 1080p video of 15 minutes duration will take approximately 20 minutes to process).
2) The system can detect a variety of different objects which are supported by the Deep Neural Network Model used. This includes a variety of objects like cars, bikes, people etc. Thus we can get the total count of cars or people in a video or the combined total of all the different kinds of objects as well.
3) The same system, with the same architecture design could be extended for different kinds of applications that involve video processing. Only the code for the worker node might change in that case, and the rest of the stack remains the same. An example could be Video Resolution Enhancement using Neural Networks.

Limitations of the System :

1) Depending on the load or the video size, we cannot automatically scale up the worker nodes. We have to manually create a new VM and add it as a worker node. This could have been avoided by using Kubernetes with automatic load balancing enabled.
2) This system relies on the fact that all the VM's remain active and do their job till the end of the processing of the video. Suppose, any one of the VM's stops working and crashes; then the frames that are processed at the time of crash would be lost. Since each VM receives a bundle of 100 frames at a time. Thus a minimum of 100 frames will be lost depending on the number of VM's that crash.