Automated Essay Scoring System with Kubernetes

Present By: Nisarg Patel

Guided By: Dr. Henry Chang

Table of Contents

Introduction

The AES system designed on two stage approach.

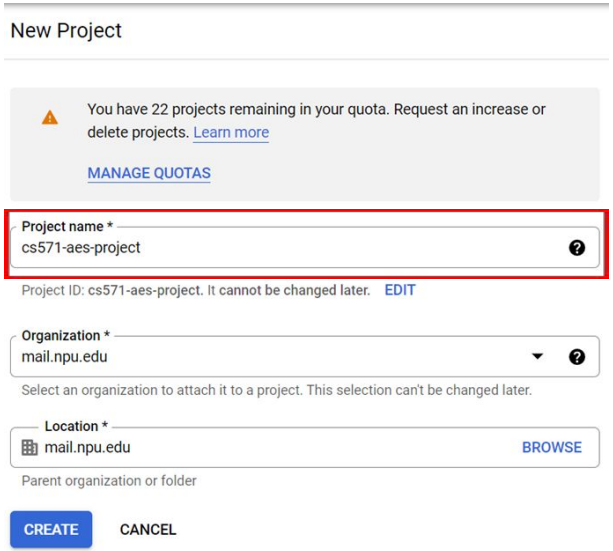
At the first deep learning model to create text-summary and second stage grading the student essay by comparing with instructor essay summary.

Built on Python Flask and PyTorch and developed by SFBU students (Zhou Quan, Thanga Rani Prabhu, Chris Zhang)

It’s running on GCP with Kubernetes deployment platform.

Design

Step 1 Create a new project on GCP for this project work.



Select the project that you created and open the GCP terminal windows.

Design

Step 2 To host the AES project on GKE cluster

Create and launch the new **aes-project** Kubernetes cluster

gcloud container clusters create aes-project --num-nodes=1 --machine-type=e2-micro --region=us-west1

```
patel19619@cloudshell:~ (cs571-aes-project)$ gcloud container clusters create aes-project --num-nodes=1 --
machine-type=e2-micro --region=us-west1
Default change: VPC-native is the default mode during cluster creation for versions greater than 1.21.0-gk
e.1500. To create advanced routes based clusters, please pass the `--no-enable-ip-alias` flag
Note: Your Pod address range (`--cluster-ipv4-cidr`) can accommodate at most 1008 node(s).
Creating cluster aes-project in us-west1... Cluster is being health-checked (master is healthy)..
.done.
Created [https://container.googleapis.com/v1/projects/cs571-aes-project/zones/us-west1/clusters/aes-projec
t].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gclo
ud/us-west1/aes-project?project=cs571-aes-project
kubeconfig entry generated for aes-project.
NAME: aes-project
LOCATION: us-west1
MASTER_VERSION: 1.21.6-gke.1503
MASTER_IP: 34.145.112.181
MACHINE_TYPE: e2-micro
NODE_VERSION: 1.21.6-gke.1503
NUM_NODES: 3
STATUS: RUNNING
```

Setup Virtual Environment

Download the latest Anaconda Miniconda shell script

wget

https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh

```
(base) patel19619@cloudshell:~/final_project$ wget https://repo.anaconda.com/mi
nda3-latest-Linux-x86_64.sh
--2022-04-06 03:34:16--  https://repo.anaconda.com/miniconda/Miniconda3-latest-
h
Resolving repo.anaconda.com (repo.anaconda.com)... 104.16.131.3, 104.16.130.3,
0:8203, ...
Connecting to repo.anaconda.com (repo.anaconda.com)|104.16.131.3|:443... connec
HTTP request sent, awaiting response... 200 OK
Length: 75660608 (72M) [application/x-sh]
Saving to: 'Miniconda3-latest-Linux-x86_64.sh.1'

Miniconda3-latest-Linux 100%[===========================>]  72.16M   165MB/s
```

Execute the miniconda installation script

chmod +x Miniconda3-latest-Linux-x86_64.sh

./Miniconda3-latest-Linux-x86_64.sh

```
(base) patel19619@cloudshell:~/final_project$ ./Miniconda3-latest-Linux-x86_64.sh
Welcome to Miniconda3 py39_4.11.0

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>>
==================================
End User License Agreement - Miniconda
==================================

Copyright 2015-2021, Anaconda, Inc.

All rights reserved under the 3-clause BSD License:
```

Activate a Virtual Environment

Create and activate a Python environment

conda create -n myenv python=3.6

```
(base) patel19619@cloudshell:~/final_project$ conda create -n myenv python=3.6
WARNING: A conda environment already exists at '/home/patel19619/miniconda3/envs/myenv'
Remove existing environment (y/[n])? y

Collecting package metadata (current_repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.11.0
  latest version: 4.12.0

Please update conda by running

    $ conda update -n base -c defaults conda
```

conda activate myenv

```
(base) patel19619@cloudshell:~/final_project$ conda activate myenv
(myenv) patel19619@cloudshell:~/final_project$
```

Integration of AES system

Clone the SFBU students AES Github Project

```
git clone https://github.com/Quan25/flask-summary.git aes_project (directory name)
```

Download rough.zip dependency file

```
wget --load-cookies /tmp/cookies.txt
"https://docs.google.com/uc?export=download&confirm=$(wget --quiet
--savecookies /tmp/cookies.txt --keep-session-cookies --no-check-certificate
'https://drive.google.com/file/d/1RxfZOYyNvzvCf37_vABfJMkohAsEZKtH/' -O- | sed -
rn 's/.confirm=([0-9A-Za-z_]+)./\1\n/p')&id=1RxfZOYyNvzvCf37_vABfJMkohAsEZKtH"
-O rough.zip && rm -rf /tmp/cookies.txt
```

Download Python Dependencies

Extract the rough.zip file

```
unzip rough.zip
```

Download the following dependencies

```
sudo apt-get install libxml-parser-perl

sudo cpan install XML::Parser::PerlSAX

sudo cpan install XML::RegExp

sudo cpan install XML::DOM

./runROUGE-test.pl
```

Download Python Dependencies

Install pyrouge library

git clone https://github.com/bheinzerling/pyrouge.git

cd pyrouge

pip3 install -e .

```
(myenv) patel19619@cloudshell:~/.../RELEASE-1.5.5/pyrouge$ pip install -e .
Obtaining file:///home/patel19619/final_project/aes_project/RELEASE-1.5.5/pyrouge
Installing collected packages: pyrouge
  Running setup.py develop for pyrouge
Successfully installed pyrouge-0.1.3
```

Integration Of Deep Learning Model

Download the pertain-bert-model on Home directory

wget https://s3.amazonaws.com/models.huggingface.co/bert/bert-large-uncased.tar.gz

Change the path on BertParent.py file

```
# from
  self.model = BertModel.from_pretrained('/home/quan/Downloads/bert-large-uncased')
# to
   self.model = BertModel.from_pretrained('/home/your_username/bert-large-uncased.tar.gz')
```

```
self.model = BertModel.from_pretrained('/home/patel19619/final_project/
aes_project/bert-large-uncased.tar.gz')
self.tokenizer = self.token_handler[model_type].from_pretrained(self.
```

Download Python Dependencies

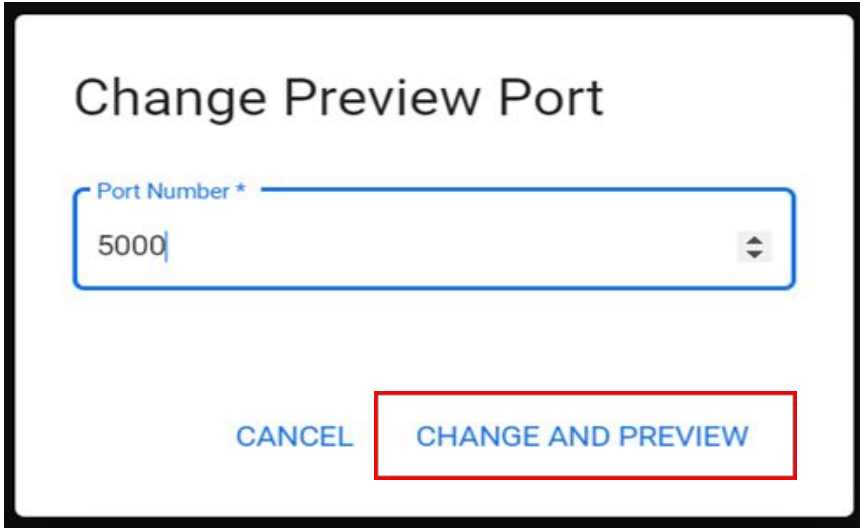
```
pip3 install torch torchvision torchaudio --extra-index-url
https://download.pytorch.org/whl/cpu
pip3 install flask pandas sklearn nltk
pip3 install gensim==3.8.3
pip3 install pytorch-pretrained-bert
pip3 install matplotlib==3.0.1

$ python3

   >import nltk

   >nltk.download('punkt')
```

Run AES system on local machine

python3 app.py

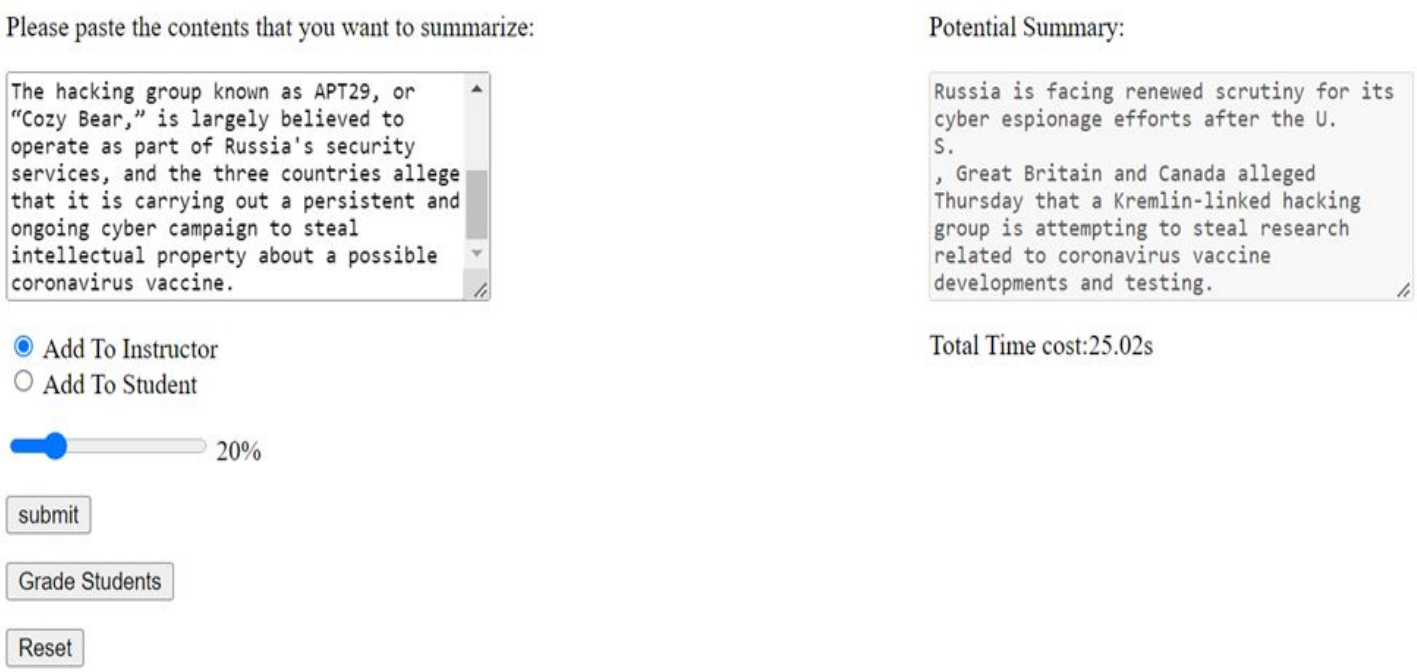
```
(myenv) patel19619@cloudshell:~/final_proje
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. D
   Use a production WSGI server instead.
 * Debug mode: on
WARNING:werkzeug: * Running on all addresse
   WARNING: This is a development server. D
INFO:werkzeug: * Running on http://172.17.0
INFO:werkzeug: * Restarting with stat
WARNING:werkzeug: * Debugger is active!
INFO:werkzeug: * Debugger PIN: 230-025-357
```

Use Web Preview and change the port to 5000



Run AES System on local machine

Add text on the to text box and select to summarize as Instructor/Student and press submit



Configuration

Hosting the AES system with Kubernetes and docker.

Create the Docker hub repository

Create a Docker file

```
FROM ubuntu:20.04

ENV TZ=America/Los_Angeles
RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
RUN mkdir -p /home/project/

COPY rough.zip /home/project/
COPY punktDownload.py /home/project/
COPY bert-large-uncased.tar.gz /home/project/

WORKDIR /home/project/
RUN apt-get update -y
RUN apt-get upgrade -y
RUN apt-get install -y build-essential python3-pip libxml-parser-perl unzip git
RUN apt-get install -y pkg-config libpng-dev libfreetype6-dev freetype2-demos

RUN unzip rough.zip
RUN cpan install XML::Parser::PerlSAX
RUN cpan install XML::RegExp
RUN cpan install XML::DOM

WORKDIR /home/project/RELEASE-1.5.5
RUN ./runROUGE-test.pl
```

Build the docker image

```
patel19619@instance-1:~$ sudo docker build -t cs571/aes .
Sending build context to Docker daemon  2.932GB
Step 1/33 : FROM ubuntu:20.04
 ---> 825d55fb6340
Step 2/33 : ENV TZ=America/Los_Angeles
 ---> Using cache
 ---> 82dcb7b71838
Step 3/33 : RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
 ---> Using cache
 ---> 0f6154791a0b
Step 4/33 : RUN mkdir -p /home/project/
 ---> Using cache
 ---> 0bd0b11801bf
Step 5/33 : COPY rough.zip /home/project/
 ---> 0aaff57fd5b9
Step 6/33 : COPY punktDownload.py /home/project/
 ---> 85e1be8ba01d
Step 7/33 : COPY bert-large-uncased.tar.gz /home/project/
```

Configuration

Push image to Docker hub

Run the created docker image

docker run -p 5000:5000 -t your_dockerhub_ID/repository_name

```
patel19619@instance-1:~$ docker run -p 5000:5000 -t cs571/aes
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
INFO:werkzeug: * Running on all addresses (0.0.0.0)
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.2:5000 (Press CTRL+C to quit)
INFO:werkzeug: * Restarting with stat
WARNING:werkzeug: * Debugger is active!
INFO:werkzeug: * Debugger PIN: 514-278-626
```

Configuration on Kubernetes

To host the system on GCP for users

minikube start

```
patel19619@cloudshell:~/aes-project$ minikube start
😄  minikube v1.25.2 on Debian 11.2 (amd64)
    ▪ MINIKUBE_FORCE_SYSTEMD=true
    ▪ MINIKUBE_HOME=/google/minikube
    ▪ MINIKUBE_WANTUPDATENOTIFICATION=false
✨  Automatically selected the docker driver. Other choices:
👍  Starting control plane node minikube in cluster minikube
🚜  Pulling base image ...
💾  Downloading Kubernetes v1.23.3 preload ...
    > preloaded-images-k8s-v17-v1...: 505.68 MiB / 505.68 M
🔥  Creating docker container (CPUs=2, Memory=4000MB) ...
🐳  Preparing Kubernetes v1.23.3 on Docker 20.10.12 ...
    ▪ kubelet.cgroups-per-qos=false
    ▪ kubelet.enforce-node-allocatable=""
    ▪ kubelet.housekeeping-interval=5m
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v
🌟  Enabled addons: storage-provisioner, default-storageclas
🏄  Done! kubectl is now configured to use "minikube" cluste
```

minikube addons enable ingress

```
| freshpod                | minikube | disabled        | google                      |
| gcp-auth                | minikube | disabled        | google                      |
| gvisor                  | minikube | disabled        | google                      |
| helm-tiller             | minikube | disabled        | third-party (helm)          |
| ingress                 | minikube | enabled ✅      | unknown (third-party)       |
| ingress-dns             | minikube | disabled        | google                      |
| istio                   | minikube | disabled        | third-party (istio)         |
| istio-provisioner       | minikube | disabled        | third-party (istio)         |
| kong                    | minikube | disabled        | third-party (Kong HQ)       |
```

Configuration on Kubernetes

Create the deployment and service yaml file

vim aes-project-deployment.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aes-deployment
spec:
  selector:
    matchLabels:
      app: aes-deployment
  replicas: 1
  template:
    metadata:
      labels:
        app: aes-deployment
    spec:
      containers:
      - name: aes-deployment
        image: cs571/aes
        ports:
        - containerPort: 5000
```

vim aes-project-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: aes-service
spec:
  selector:
    app: aes-deployment
  ports:
  - protocol: TCP
    port: 5000
    targetPort: 5000
```

Configuration on Kubernetes

vim aes-ingress.yaml

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: aes-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /$2
spec:
  rules:
    - host: cs571.aesproject.com
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: aes-service
                port:
                  number: 5000
```

Build the deployment, service and ingress file

kubectl apply -f aes-deployment.yaml

kubectl apply -f aes-service.yaml

kubectl apply -f aes-ingress.yaml

Kubectl get ingress

```
patel19619@cloudshell:~/aes-project$ kubectl get ingress
NAME          CLASS   HOSTS              ADDRESS        PORTS   AGE
aes-ingress   nginx   aesproject.com   192.168.49.2   80      118s
```

Configuration on Kubernetes

Add the IP address mentioned on get ingress result to end of the file

vi /etc/hosts

```
# Kubernetes-managed hosts file.
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loo
fe00::0 ip6-localnet
fe00::0 ip6-mcastprefix
fe00::1 ip6-allnodes
fe00::2 ip6-allrouters
172.17.0.4      cs-810844977107-default
192.168.49.2  cs571.aesproject.com
```

kubectl get all

```
patel19619@cloudshell:~/aes-project$ kubectl get all
NAME                                  READY   STATUS    RESTARTS   AGE
pod/aes-deployment-575cffc449-5hmbm   1/1     Running   0          5m47s

NAME                  TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
service/aes-service   ClusterIP   10.111.168.108   <none>        5000/TCP   5m29s
service/kubernetes    ClusterIP   10.96.0.1        <none>        443/TCP    46m

NAME                              READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/aes-deployment    1/1     1            1           5m47s

NAME                                        DESIRED   CURRENT   READY   AGE
replicaset.apps/aes-deployment-575cffc449   1         1         1       5m47s
```

Run the Application with Domain name

Go to Web browser and search **cs571.aesproject.com**