UCLA Samueli
Computer Science

CS145: Introduction to Data Mining (Spring 2024)

Discussion 4: **Project Baselines**

Instructor: Dr. Ziniu Hu
Teaching Assistant: Yanqiao Zhu

The Scalable Analytics Institute (ScAI)
Department of Computer Science
University of California, Los Angeles (UCLA)

Announcements

1. HW1 scores have been released
2. HW2 solution has been released
3. HW3 due next Monday (April 29)
4. HW4 will be released this weekend
5. Google Cloud credit released; redeem it ASAP
6. New deadline: Project proposal due on May 13

Project proposal

- Submit by May 13, 11:59 PM
- One submission per team (include all members)
- Use NeurIPS LaTeX style files: 2 pages max (excluding references)
- Include:
  - Problem statement
  - Literature review
  - Tentative schedule
  - Tentative approach
  - Division of workload per member
  - References (if any)

Project proposal

- Run the official baselines
- Survey literature for improvement ideas
- Propose ≥1 method to improve baseline
- Discuss with TA/professor to formalize idea
- Use proposal as blueprint for final report

Recap: K-means

- Goal: Partition n data points into K clusters, minimizing the Within-Cluster Sum of Squares (WCSS)
- WCSS measures the compactness of clusters by summing the squared distances between data points and their assigned centroids

$$\mathrm{WCSS} = \sum_{k=1}^{K} \sum_{\boldsymbol{x}_i \in C_k} \|\boldsymbol{x}_i - \boldsymbol{c}_k\|^2$$

  - $\mathbf{x}_i$: The i-th data point in the dataset
  - $\mathbf{c}_k$: The centroid of the k-th cluster
  - C_k: The set of data points assigned to the k-th cluster
  - $|C_k|$: The number of data points in the k-th cluster

Recap: K-means

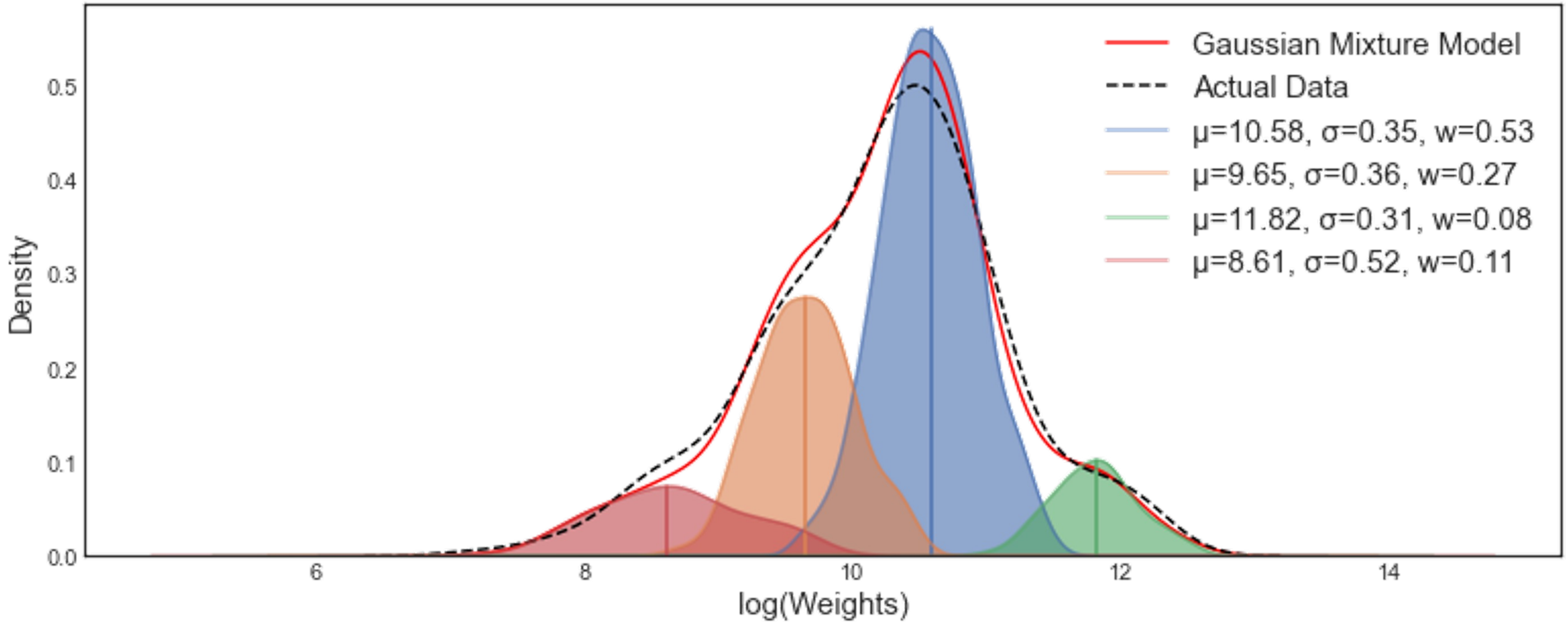
- How does the objective WCSS relate to the assignment and update steps in K-means?
  - The assignment step minimizes WCSS by assigning each data point to the nearest centroid
$$\arg\min_k \|\boldsymbol{x}_i - \boldsymbol{c}_k\|^2$$
  - The update step minimizes WCSS by recalculating centroids as the mean of assigned data points
$$\boldsymbol{c}_k = \frac{1}{|C_k|} \sum_{\boldsymbol{x}_i \in C_k} \boldsymbol{x}_i$$
  - The assignment and update steps iteratively optimize WCSS until convergence

Recap: From K-means to GMM

- K-means is a hard clustering algorithm, where each data point is assigned to exactly one cluster
- Gaussian Mixture Models (GMM) extend the idea of K-means by introducing a probabilistic framework
- In GMM, each cluster is represented by a Gaussian distribution, characterized by its mean, covariance matrix, and mixing coefficient
- The goal of GMM is to model the data as a mixture of K Gaussian distributions

Recap: From K-means to GMM

- In GMM, each cluster is represented by a Gaussian distribution, characterized by its mean, covariance matrix, and mixing coefficient



Recap: GMM formulation

- What does each parameter in the GMM formulation represent?
  - GMM models the probability density function of data as a weighted sum of Gaussian distributions

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

  - π_k: Mixing coefficient (prior probability) of the k-th component, where $\sum_{k=1}^{K} \pi_k = 1$
  - $\boldsymbol{\mu}_k$: Mean vector of the k-th Gaussian component
  - $\boldsymbol{\Sigma}_k$: Covariance matrix of the k-th Gaussian component

Recap: GMM formulation

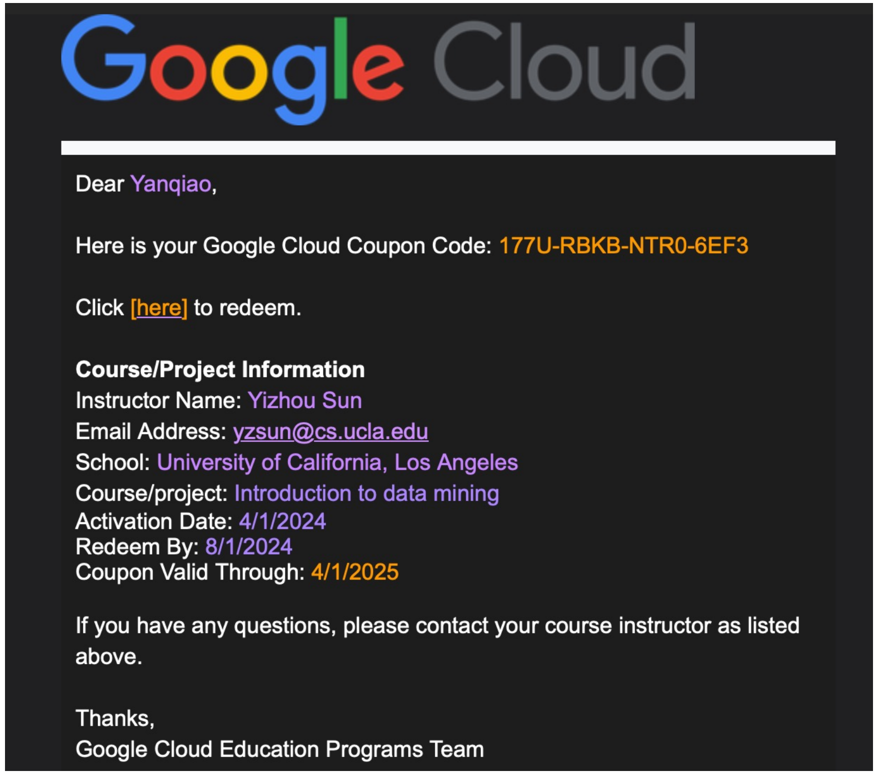
- What are the prior and posterior distributions in GMM?
  - Prior distribution:
    - The prior distribution over the latent variables $\boldsymbol{z}_i$ is a categorical distribution parameterized by the mixing coefficients π_k
    - $p(\boldsymbol{z}_i \mid \boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{z_{ik}}$
  - Posterior distribution:
    - The posterior distribution is the probability of a data point belonging to each component given the observed data and current parameters
    - It is computed in the E-step of the EM algorithm using Bayes' theorem
    - $\gamma(z_{ik}) = p(z_{ik} = 1 \mid \boldsymbol{x}_i, \boldsymbol{\theta}^{\text{old}}) = \frac{\pi_k^{\text{old}} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k^{\text{old}}, \boldsymbol{\Sigma}_k^{\text{old}})}{\sum_{j=1}^{K} \pi_j^{\text{old}} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j^{\text{old}}, \boldsymbol{\Sigma}_j^{\text{old}})}$

Recap: GMM optimization

- Why and how is the EM algorithm used to optimize GMM parameters?
  - GMM parameters cannot be directly optimized using gradient-based methods due to latent variables and non-convexity
  - The EM algorithm is used to iteratively optimize the GMM parameters by alternating between two steps:
    - E-step (Expectation): Compute the posterior probabilities (responsibilities) of each data point belonging to each component.
    - M-step (Maximization): Update the parameters (mixing coefficients, means, and covariances) using the computed responsibilities.
  - The E-step estimates the latent variables given the current parameters, while the M-step updates the parameters based on the estimated latent variables

Using Google cloud servers

- Verify your UCLA email to receive the coupon
- Create a Google Cloud account
- Redeem your coupon using this link



Google Cloud

Dear Yanqiao,

Here is your Google Cloud Coupon Code: 177U-RBKB-NTR0-6EF3

Click [here] to redeem.

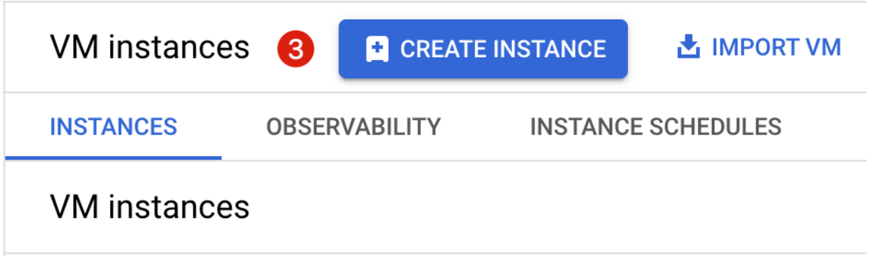
Course/Project Information
Instructor Name: Yizhou Sun
Email Address: yzsun@cs.ucla.edu
School: University of California, Los Angeles
Course/project: Introduction to data mining
Activation Date: 4/1/2024
Redeem By: 8/1/2024
Coupon Valid Through: 4/1/2025

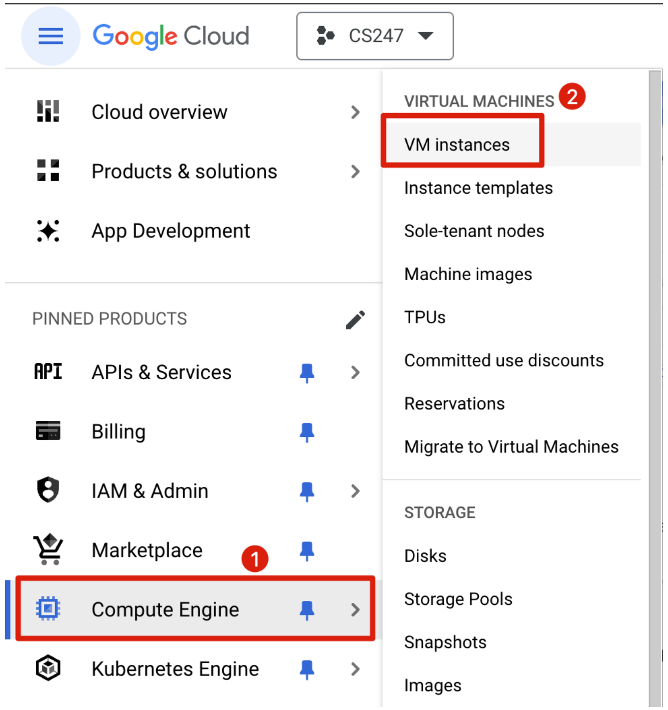
If you have any questions, please contact your course instructor as listed above.

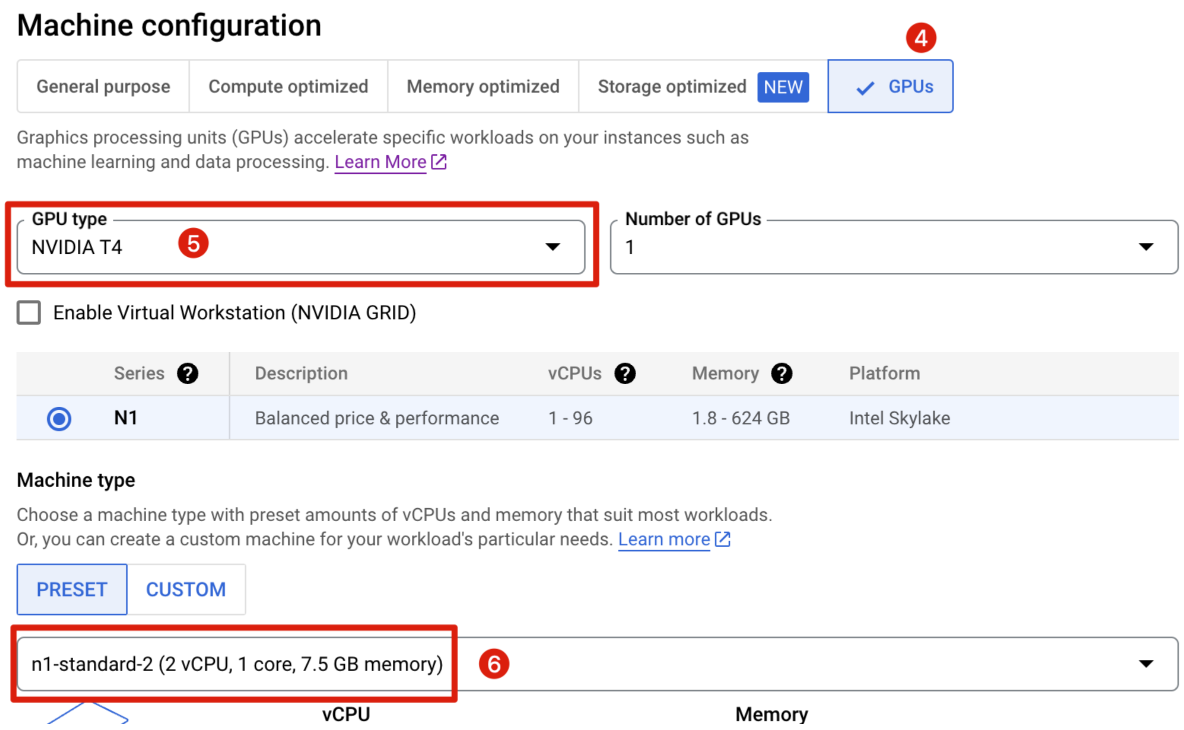
Thanks,
Google Cloud Education Programs Team

Using Google cloud servers

- Create a new VM instance
- Choose configurations that you like

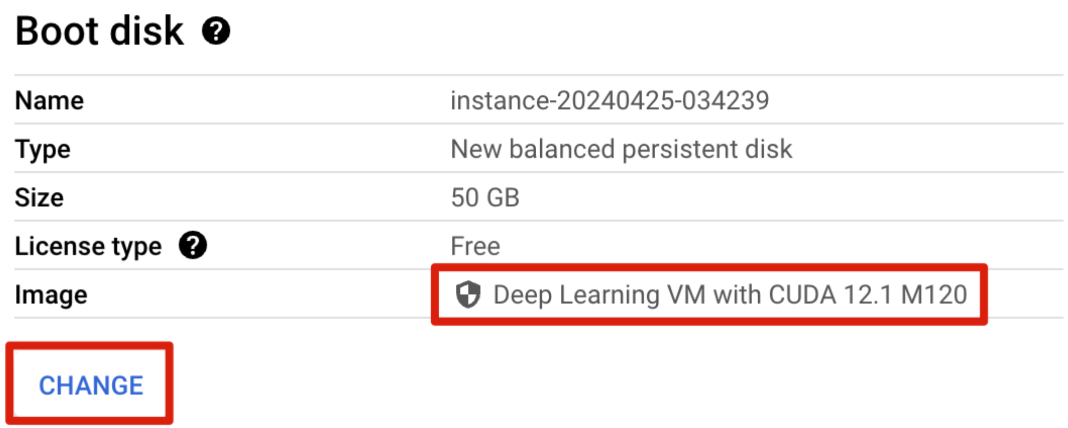






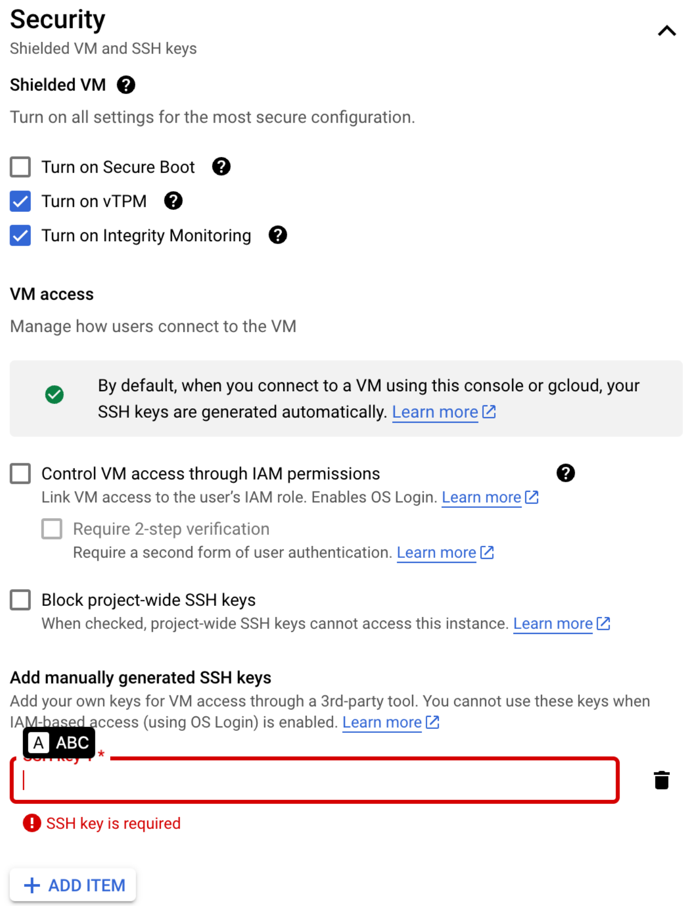
Using Google cloud servers

- Make sure the boot image option is selected to Deep Learning VM
- Otherwise, you will need to install CUDA by yourself



Using Google cloud servers

- If you haven't got your SSH keys, check this tutorial to create an SSH key pair
- Provide your public SSH keys for login



Using Google cloud servers

- After creating the instance, you can access it via the console

Filter Enter property name or value

	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP	Connect	
☑	✔	instance-20240425-034239	us-west4-a			10.182.0.3 (nic0)	34.16.208.137 (nic0)	SSH ▾	⋮

```
======================================
Welcome to the Google Deep Learning VM
======================================

Version: common-cu121.m120
Resources:
 * Google Deep Learning Platform StackOverflow: https://stackoverflow.com/questions/tagged/google-dl-platform
 * Google Cloud Documentation: https://cloud.google.com/deep-learning-vm
 * Google Group: https://groups.google.com/forum/#!forum/google-dl-platform

To reinstall Nvidia driver (if needed) run:
sudo /opt/deeplearning/install-driver.sh
Linux instance-20240425-034239 5.10.0-28-cloud-amd64 #1 SMP Debian 5.10.209-2 (2024-01-31) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

This VM requires Nvidia drivers to function correctly.   Installation takes ~1 minute.
Would you like to install the Nvidia driver? [y/n] y
```

Using Google cloud servers

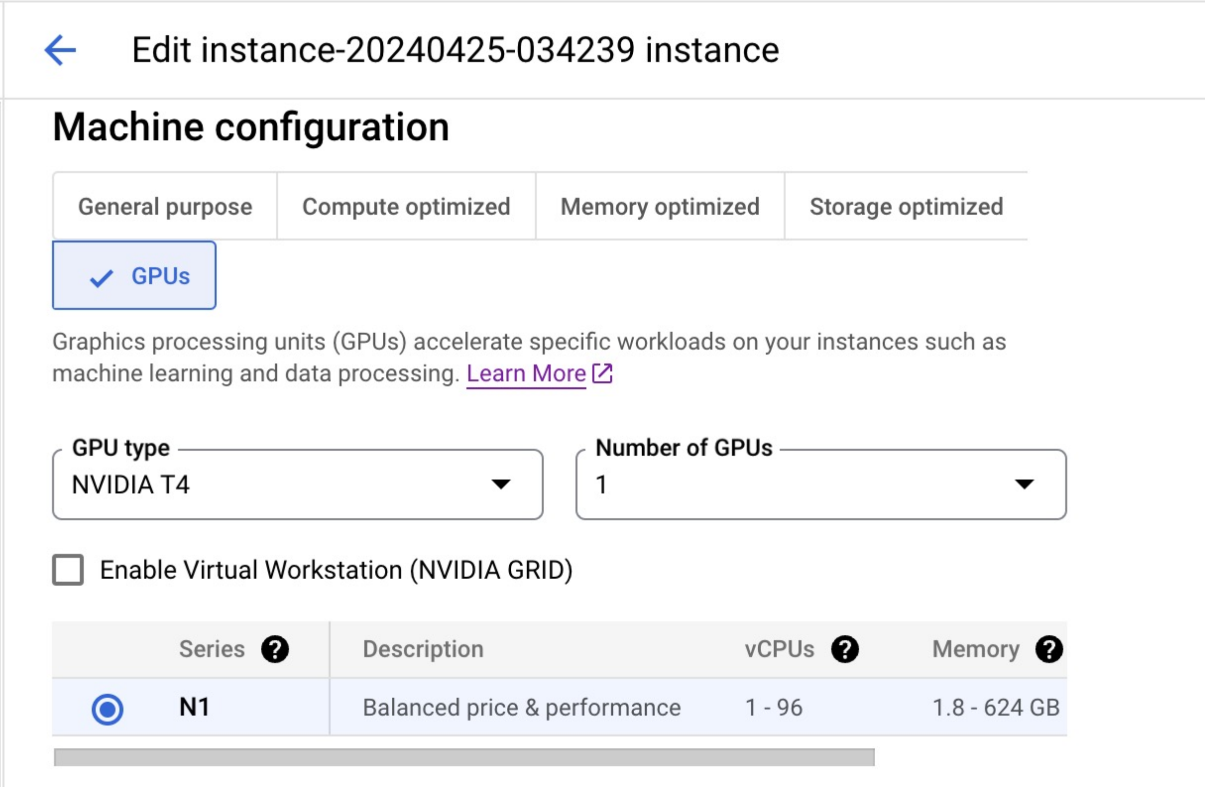
- Check the GPU usage via nvidia-smi

```
(base) mr_sxkdz@instance-20240425-034239:~$ nvidia-smi
Thu Apr 25 03:51:52 2024
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.86.10              Driver Version: 535.86.10    CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|=========================================+======================+======================|
|   0  Tesla T4                       Off | 00000000:00:04.0 Off |                    0 |
| N/A   76C    P0              35W /  70W |      2MiB / 15360MiB |      7%      Default |
|                                         |                      |                  N/A |
+-----------------------------------------+----------------------+----------------------+

+---------------------------------------------------------------------------------------+
| Processes:                                                                            |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=======================================================================================|
|  No running processes found                                                           |
+---------------------------------------------------------------------------------------+
```

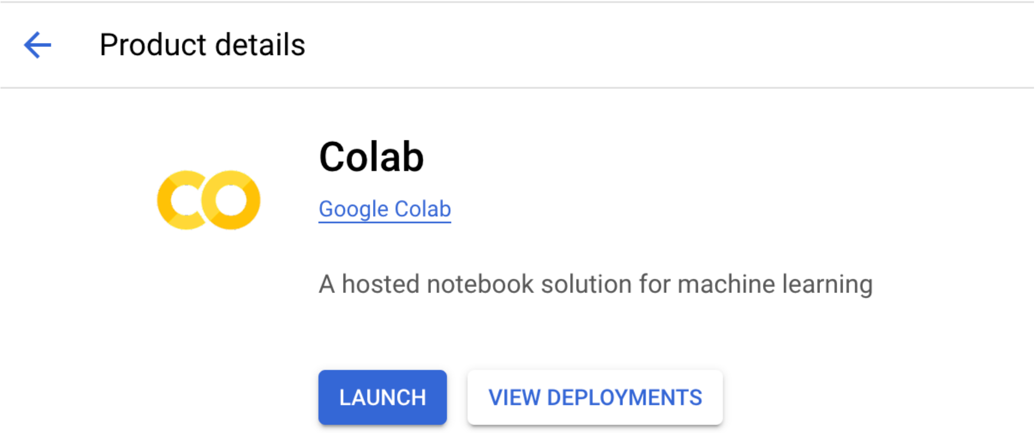
Using Google cloud servers

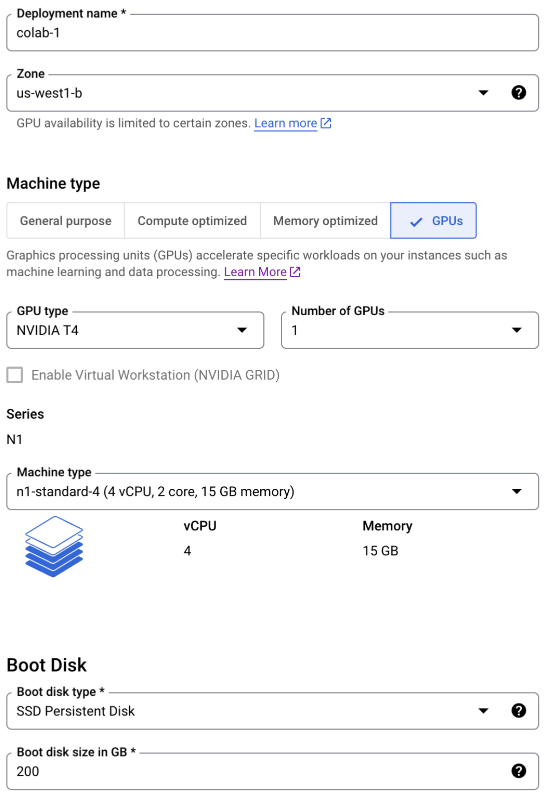
- Upgrade/downgrade server configurations after stopping the VM



Using Google cloud servers

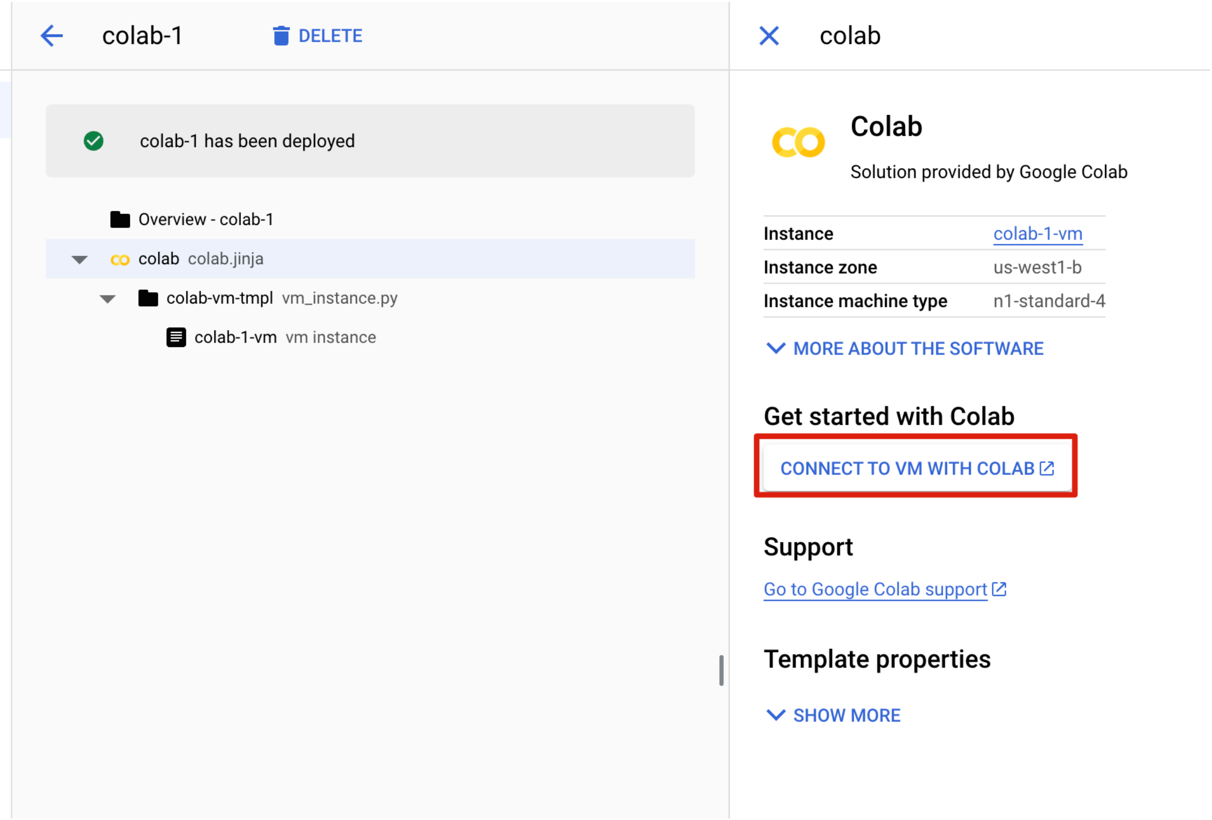
- You can also launch Colab to avoid cumbersome terminal interactions

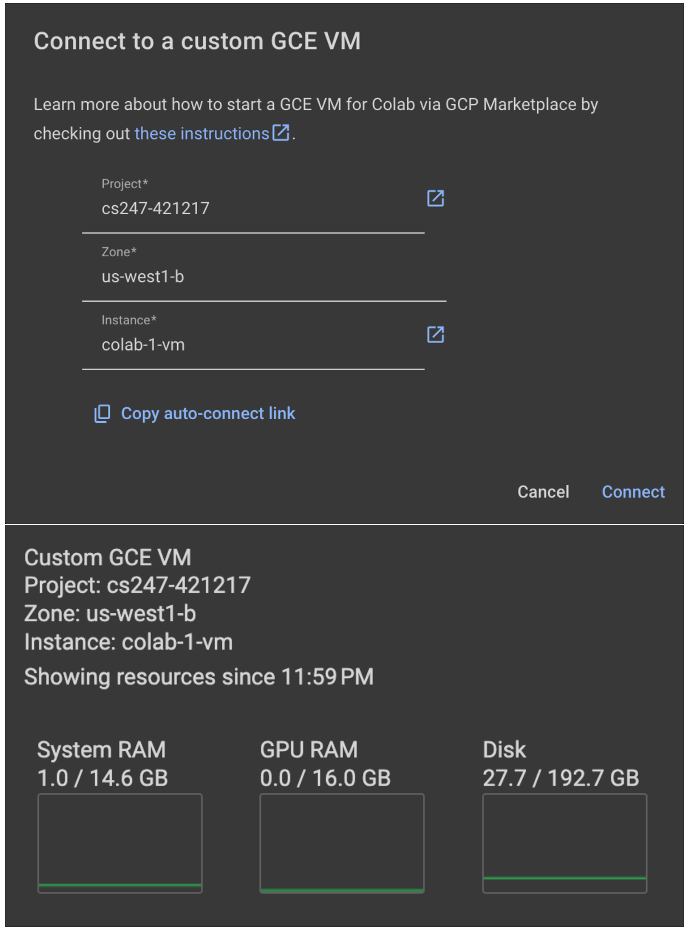




Using Google cloud servers

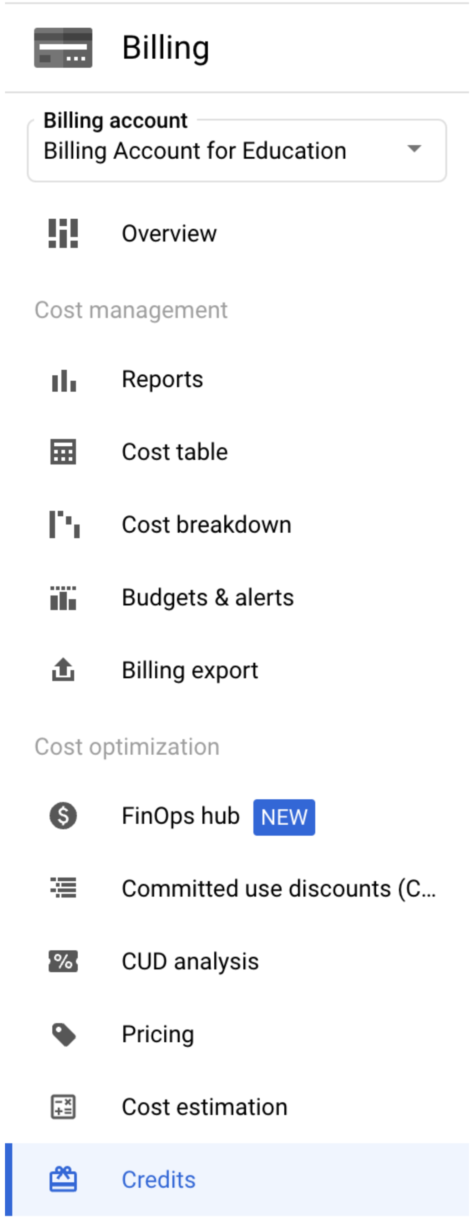
- Connect to Google Cloud after launching Colab





Using Google cloud servers

- Create a VM without GPUs at first to process datasets and save your wallet
- Move to cloud servers when you are ready to train your model
- Remember to shut down your VM if you finished training
- Watch out your remaining balance carefully



Run baselines: IND

- Download the dataset
  - `wget https://www.dropbox.com/scl/fi/o8du146aafl3vrb87tm45/IND-WhoIsWho.zip?rlkey=cg6tbubqo532hb1ljaz70tlxe&dl=1`
  - `unzip IND-WhoIsWho.zip`
- Clone the baseline repository
  - `git clone https://github.com/THUDM/whoiswho-top-solutions.git`
  - `cd whoiswho-top-solutions/incorrect_assignment_detection`
- Install required packages
  - `pip install -r requirements.txt`

Run baselines: IND

- Preprocess data
  - `python encoding.py --path pid_to_info_all.json --save_path roberta_embeddings.pkl`
  - `python build_graph.py --author_dir train_author.json --save_dir train.pkl --pub_dir pid_to_info_all.json --embeddings_dir roberta_embeddings.pkl`
  - `python build_graph.py --author_dir ind_valid_author.json --save_dir valid.pkl --pub_dir pid_to_info_all.json --embeddings_dir roberta_embeddings.pkl`
- Train & test the model
  - `python train.py --train_dir train.pkl --test_dir valid.pkl`

Run baselines: PST

- Download the dataset
  - `wget https://www.dropbox.com/scl/fi/namx1n55xzqil4zbkd5sv/PST.zip?rlkey=impcbm2acqmqhurv2oj0xxysx&dl=1`
  - `unzip PST.zip`
  - `wget https://opendata.aminer.cn/dataset/DBLP-Citation-network-V16.zip`
  - `unzip DBLP-Citation-network-V16.zip`
- Put the unzipped PST directory into `data/` and unzipped DBLP dataset into `data/PST/`
- Clone the baseline repository
  - `git clone https://github.com/THUDM/paper-source-trace.git`
  - `cd paper-source-trace`

Run baselines: PST

- Install required packages
  - `pip install -r requirements.txt`
- Run baselines
  - `# Method 1: Random Forest`
  - `python rf/process_kddcup_data.py`
  - `python rf/model_rf.py  # output at out/kddcup/rf/`

  - `# Method 2: Network Embedding`
  - `python net_emb.py  # output at out/kddcup/prone/`

  - `# Method 3: SciBERT`
  - `python bert.py  # output at out/kddcup/scibert/`