

submit the github directory link with all the details in a nice readme file.

this is an individual assignment

PROJECT REPORT

Create a public github directory and add all your artifacts - slide deck, project artifacts screenshots, presentation (all team members video presentation) and all code artifacts (as well as recording of your project presentation - long version) links and submit the github link - the github directory should have a proper MUST HAVE read.me file highlighting all the work and summary in nicely organized fashion with links to deliverables..

Explain very very clearly the project team what they did other than the github code or existing colab on web - highlight also each team member contribution. gradio ux is minimum needed - otherwise you can have a full web server ux.

also update the github link in the spreadsheet.

Clearly document in proposal what your end deliverables will be.

More guidance on how to approach the project in this [link](#)

(<http://cs231n.stanford.edu/project.html>)..


<https://github.com/visenger/awesome-mlops> (<https://github.com/visenger/awesome-mlops>)

The end deliverables of the project are

It is strongly preferable to do project on sagemaker, huggingfacespaces or vertex ai or any other good reputed startup end2end mlops or azure end2end mlops with all artifacts or data bricks

Databricks has a nice course on end2end mlops - please take that 300 dollars credit course and if you put your project thru data bricks pipeline that will be great easy way to get 10% extra points (with all artifacts for auto retrain and auto deploy) = you need to demo these extra artifacts of auto push and auto deploy to get the extra 10% points.

Colab is least preferred option (you will score less points)


Make sure you can do vibe coding (see <https://www.deeplearning.ai/short-courses/vibe-coding-101-with-replit/> ) (<https://www.deeplearning.ai/short-courses/vibe-coding-101-with-replit/>) - so you can generate complex ux and complex models with all artifacts for ml ops.

a) entire application archive or Colab with execution on git hub and all the artifacts -screenshots etc., - aka not just colab - the actual execution summary and results in addition. The colab should be runnable (aka retrain and infer). Production demo of website doing inference. Production pipeline of the same using any cloud providers showing model training, eval etc.,. Proper eval dashboards (tensorboard) and other artifacts. Caution: Turnitin will be used heavily for checking. copy pasting existing colabs content available in google web search is a BIG no no. The expectation is for students to actually write the project end2end from scratch.

b) 20% of the project should show case all the visualization techniques of model metrics. (very important) - what are the core metrics for the ml model and how they fare for the model you trained. proper principles of data set split, data augmentation etc.,. should all be applied

c) colab/project should be very heavily documented. Also section in colab highlighting why we used what type of parameters like loss functions, activation functions, normalization, augmentation etc.,. should be present. readme.md is must

d) do not use any reinforcement learning in your project. make sure you have very good data set (either you do it synthetically or manually curate it or get it from roboflow annotations or get any benchmark data sets as last resort)

if you want, you can use <https://openai.com/index/paperbench>  (<https://openai.com/index/paperbench>) for replicating any existing paper code into your application.

e) team size and team members and each team member contributions etc.,. mentioned clearly.

f) what are the inputs, outputs, what are the key metrics to evaluate the model should be elaborated.

g) The core defining the model - code should be substantial (if the model definition is just 10 - 50 lines - something wrong - 50% of score is based on how complex the problem you are solving and how complex your model or combination of models are) and is the meat of the project. Your model architecture, training loop, etc.,.

Do not try to risk having projects requiring large number of gpus/tpus - show case it on very small data set. extra points if you train your project on tpu.

Follow MLOps practices (similar to devops) - <https://github.com/microsoft/MLOps> (<https://github.com/microsoft/MLOps>) - <https://towardsdatascience.com/the-rise-of-the-term-mlops-3b14d5bd1bdb> (<https://towardsdatascience.com/the-rise-of-the-term-mlops-3b14d5bd1bdb>) - if in doubt refer to

<https://github.com/GokuMohandas/mlops-course> 

(<https://github.com/GokuMohandas/mlops-course>) - make sure your project artifacts are complete as in this course for an extra 10 % points with full mlops maturity level 4 see below.

(<https://github.com/GokuMohandas/mlops-course>) for artifacts

or

10% extra credit of course - If you use Full Stack Deep Learning with special emphasis on ci/cd, auto deployment, metrics and monitoring and drift detection and mitigation -this need to be showcased fully in demo the mlops maturity level 4 - <https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model> (<https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model>) - <https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning> (<https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning>) - otherwise no 10% given - demo all the work you did on full stack production quality ml ops maturity level 2 deep learning project - see for example: <https://github.com/GoogleCloudPlatform/mlops-with-vertex-ai> (<https://github.com/GoogleCloudPlatform/mlops-with-vertex-ai>)

<https://github.com/GoogleCloudPlatform/vertex-pipelines-end-to-end-samples> (<https://github.com/GoogleCloudPlatform/vertex-pipelines-end-to-end-samples>)

<https://github.com/aws-samples/mlops-amazon-sagemaker> (<https://github.com/aws-samples/mlops-amazon-sagemaker>)

<https://github.com/aws-samples/mlops-e2e>

<https://learn.microsoft.com/en-us/training/paths/build-first-machine-operations-workflow> (<https://learn.microsoft.com/en-us/training/paths/build-first-machine-operations-workflow>)

https://catalog.us-east-1.prod.workshops.aws/workshops/44d3e2a0-ec6f-44df-9397-bcfd129cadf/en-US/module_introduction_1 (https://catalog.us-east-1.prod.workshops.aws/workshops/44d3e2a0-ec6f-44df-9397-bcfd129cadf/en-US/module_introduction_1)

<https://www.databricks.com/resources/demos/tutorials/data-science-and-ai/mlops-end-to-end-pipeline> (<https://www.databricks.com/resources/demos/tutorials/data-science-and-ai/mlops-end-to-end-pipeline>)

<https://cloud.google.com/architecture/architecture-for-mlops-using-tfx-kubeflow-pipelines-and-cloud-build?hl=en>

<https://github.com/renardeinside/e2e-mlops-demo> (<https://github.com/renardeinside/e2e-mlops-demo>)

<https://docs.databricks.com/en/mlflow/end-to-end-example.html>

(<https://docs.databricks.com/en/mlflow/end-to-end-example.html>)

The MLOps maturity model encompasses five levels of technical capability:

Level	Description	Highlights	Technology
0	No MLOps https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model#level-0-no-mlops	<ul style="list-style-type: none"> • Difficult to manage full machine learning model lifecycle • The teams are disparate and releases are painful • Most systems exist as "black boxes," little feedback during/post deployment 	<ul style="list-style-type: none"> • Manual builds and deployments • Manual testing of model and application • No centralized tracking of model performance • Training of model is manual
1	DevOps but no MLOps https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model#level-1-devops-no-mlops	<ul style="list-style-type: none"> • Releases are less painful than No MLOps, but rely on Data Team for every new model • Still limited feedback on how well a model performs in production • Difficult to trace/reproduce results 	<ul style="list-style-type: none"> • Automated builds • Automated tests for application code
2	Automated Training https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model#level-2-automated-training	<ul style="list-style-type: none"> • Training environment is fully managed and traceable • Easy to reproduce model • Releases are manual, but low friction 	<ul style="list-style-type: none"> • Automated model training • Centralized tracking of model training performance • Model management

Level	Description	Highlights	Technology
3	Automated Model Deployment (https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model#level-3-automated-model-deployment)	<ul style="list-style-type: none"> Releases are low friction and automatic Full traceability from deployment back to original data Entire environment managed: train > test > production 	<ul style="list-style-type: none"> Integrated A/B testing of model performance for deployment Automated tests for all code Centralized tracking of model training performance
4	Full MLOps Automated Operations (https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model#level-4-full-mlops-automated-retraining)	<ul style="list-style-type: none"> Full system automated and easily monitored Production systems are providing information on how to improve and, in some cases, automatically improve with new models Approaching a zero-downtime system 	<ul style="list-style-type: none"> Automated model training and testing Verbose, centralized metrics from deployed model

[https://www.youtube.com/watch?](https://www.youtube.com/watch?v=_pLe7_b5tGc&list=PL1T8fO7ArWlCtdKakp2Wzn6ludwFvfQLp)

[v=_pLe7_b5tGc&list=PL1T8fO7ArWlCtdKakp2Wzn6ludwFvfQLp](https://www.youtube.com/watch?v=_pLe7_b5tGc&list=PL1T8fO7ArWlCtdKakp2Wzn6ludwFvfQLp)

<https://www.youtube.com/watch?v=Dk-95mt0MLA&list=PL1T8fO7ArWldeCszMMStl32fCtdPzYFJn&index=1>
[\(https://www.youtube.com/watch?v=Dk-95mt0MLA&list=PL1T8fO7ArWldeCszMMStl32fCtdPzYFJn&index=1\)](https://www.youtube.com/watch?v=Dk-95mt0MLA&list=PL1T8fO7ArWldeCszMMStl32fCtdPzYFJn&index=1)



<https://www.youtube.com/watch?v=Dk-95mt0MLA&list=PL1T8fO7ArWldeCszMMStl32fCtdPzYFJn&index=1>
[\(https://www.youtube.com/watch?v=Dk-95mt0MLA&list=PL1T8fO7ArWldeCszMMStl32fCtdPzYFJn&index=1\)](https://www.youtube.com/watch?v=Dk-95mt0MLA&list=PL1T8fO7ArWldeCszMMStl32fCtdPzYFJn&index=1)

use TFX and AI Pipeline or proper framework end2end with all artifacts - as well as CI/CD ML Ops level 2 and best practices of software engineering of ML (discussed in the lecture 1/2 slides) - for extra points - <https://www.tensorflow.org/tfx/tutorials> - [e2e example -](https://www.tensorflow.org/tfx/tutorials)

<https://www.tensorflow.org/tfx/tutorials>

https://github.com/tensorflow/tfx/blob/master/tfx/examples/chicago_taxi_pipeline/README

https://github.com/tensorflow/tfx/blob/master/tfx/examples/chicago_taxi_pipeline/README.md

(<https://www.tensorflow.org/tfx/tutorials>)

and <https://github.com/tensorflow/tfx/blob/master/docs/guide/index.md> and a nice video -

(<https://github.com/tensorflow/tfx/blob/master/docs/guide/index.md>)

[TensorFlow Extended \(TFX\) Overview and Pre-training Workflow \(TF Dev Summit '19\)](#)

([https://www.youtube.com/watch?v=A5wiwT1qFjc&list=UU0rqucBdTuFTjJiefW5t-](https://www.youtube.com/watch?v=A5wiwT1qFjc&list=UU0rqucBdTuFTjJiefW5t-IQ&index=45&app=desktop)

[IQ&index=45&app=desktop](https://www.youtube.com/watch?v=A5wiwT1qFjc&list=UU0rqucBdTuFTjJiefW5t-IQ&index=45&app=desktop))



([https://www.youtube.com/watch?v=A5wiwT1qFjc&list=UU0rqucBdTuFTjJiefW5t-](https://www.youtube.com/watch?v=A5wiwT1qFjc&list=UU0rqucBdTuFTjJiefW5t-IQ&index=45&app=desktop)

[IQ&index=45&app=desktop](https://www.youtube.com/watch?v=A5wiwT1qFjc&list=UU0rqucBdTuFTjJiefW5t-IQ&index=45&app=desktop))

(<https://github.com/tensorflow/tfx/blob/master/docs/guide/index.md>)

create a github public url of your account and paste the url in submission. make sure you can access this without login.

The project must have a model training component.

- **Title, Author(s)**
- **Abstract:** Briefly describe your problem, approach, and key results. Should be no more than 300 words.
- **Introduction (10%):** Describe the problem you are working on, why it's important, and an overview of your results
- **Related Work (10%):** Discuss published work that relates to your project. How is your approach similar or different from others?
- **Data (10%):** Describe the data you are working with for your project. What type of data is it? Where did it come from? How much data are you working with? Did you have to do any preprocessing, filtering, or other special treatment to use this data in your project?
- **Methods (30%):** Discuss your approach for solving the problems that you set up in the introduction. Why is your approach the right thing to do? Did you consider alternative approaches? You should demonstrate that you have applied ideas and skills built up during the quarter to tackling your problem of choice. It may be helpful to include figures, diagrams, or tables to describe your method or compare it with other methods.
- **Experiments (30%):** Discuss the experiments that you performed to demonstrate that your approach solves the problem. The exact experiments will vary depending on the project, but you might compare with previously published methods, perform an ablation study to determine the impact of various components of your system, experiment with different

hyperparameters or architectural choices, use visualization techniques to gain insight into how your model works, discuss common failure modes of your model, etc. You should include graphs, tables, or other figures to illustrate your experimental results.

- **Conclusion (5%)** Summarize your key results - what have you learned? Suggest ideas for future extensions or new applications of your ideas.
- **Writing / Formatting (5%)** Is your paper clearly written and nicely formatted?
- **Supplementary Material**, not counted toward your 6-8 page limit and submitted as a separate file. Your supplementary material might include:
 - Source code (if your project proposed an algorithm, or code that is relevant and important for your project.).
 - Cool videos, interactive visualizations, demos, etc.

Please submit early your proposal before the deadline and seek feedback (iterations) with me.

The project selected should be related to large language models or foundation models use and prompting and prompt engineering

All the best practices of ML Ops are applicable here -except sometimes ml model is replaced with a prompt.

This topic is closed for comments.
